INFORMATION AND COMPUTATION 91, 1–14 (1991)

Lower Bounds for Depth-Restricted Branching Programs

MATTHIAS KRAUSE

Sektion Mathematik, Bereich Berechnungstheorie,
Humboldt-Universität zu Berlin, 1086 Berlin, Germany

We present a new method for proving lower bounds on the complexity of branching programs and consider k-times-only branching programs. While exponential and nearly exponential lower bounds on the complexity of one-time-only branching programs were proved for many problems, there are still missing methods of proving lower bounds for k-times-only programs ($k > 1$). We prove exponential lower bounds for k-times-only branching programs which have the additional restriction that the input bits are read k times, yet blockwise and in each block in the same order. This is done both for the algebraic decision problem $\mathrm{POLY}^*_{n,d}$ ($n \in \mathbb{N}$ prime, $d \leqslant n$) whether a given mapping $g: \mathbb{F}_n \to \mathbb{F}_n$ is a polynomial over $\mathbb{F}_n$ of degree at most d, and for the corresponding monotone problem over quadratic Boolean matrices. As a consequence we obtain a sharp bound of order $\Theta(n \cdot \log(n))$ on the communication complexity of $\mathrm{POLY}^*_{n,\delta n}$ ($\delta \in (0, \frac{1}{2})$).

1. INTRODUCTION

One of the major goals of theoretical computer science is to investigate the computation of Boolean functions by circuitry-based models such as Boolean circuits, formulae, contact schemes, and branching programs. It seems that combinatorial and counting techniques can be applied more directly to these models than to the very complex types of Turing machines. Therefore, circuitry-based characterizations of central complexity classes such as NC^1, L, NL, P, and NP, and techniques for proving lower bounds on the complexity of circuitry-based models have become increasingly important.

It is well known that nearly all sequences of Boolean functions are computable only be exponential size circuits. However, the best known lower bounds on the complexity of unrestricted circuits are linear ones for Boolean networks [P; Bl], nearly quadratic lower bounds for contact schemes, formulae, and branching programs [N], and superquadratic lower bounds for formulae over the basis $\{ \vee , \wedge , \neg \}$ [A2].

In order to gain more insight into the problem of proving lower bounds, more restricted models have been considered such as width- or depth-restricted formulae and branching programs, or circuits and formulae over incomplete bases.

0890-5401/91 $3.00

Today we have techniques for proving exponential lower bounds for some models of circuits in which null-chains are forbidden. These models are monotone circuits [R; A1; AB], circuits over the basis $\{\wedge, \vee, \neg\}$ without null-chains [Ku], contact schemes without null-chains [J], and one-time-only branching programs [W; PZ; KW; K1; A&].

But in all these cases, exponential lower bounds can be proved for those sequences of Boolean functions which have only polynomial complexity in the corresponding general models. Especially in the case of branching programs, languages can be found which have exponential complexity in the model of one-time-only branching programs but quadratic or linear complexity even in the model of two-times-only branching programs [W; K2]. However, what is still missing are methods for proving lower bounds on the complexity of k-times-only programs which are better than those for the general model. We present a new method of proving lower bounds on the complexity of branching programs (Section 3) and apply this technique to some natural problems (Section 4). We consider, e.g., the algebraic decision problem $\mathrm{POLY}^*_{n,d}$ ($n \in \mathbb{N}$ prime, $d \leqslant n$) whether a given mapping g: $\mathbb{F}_n \to \mathbb{F}_n$ is a polynomial of degree at most d, and the corresponding monotone decision problem $\mathrm{POLY}_{n,d}$ whether a given subset Y of $\mathbb{F}_n \times \mathbb{F}_n$ contains the graph of such a polynomial. Our technique yields exponential lower bounds for the model of $(k, *)$-programs containing those programs which read the input bits k times, yet blockwise and in each block in the same order.

Properties of this restricted model of k-times-only programs will be discussed in Section 5. As in the general case, languages can be found which have exponential complexity in the model of one-time-only programs and polynomial complexity in the model of $(2, *)$-programs (cf. Remark 5.1). On the other hand, among the languages with exponential complexity in the model of $(k, *)$-programs for all $k \in \mathbb{N}$, there are examples which can be computed by linear size oblivious 2-times-only programs, as well as languages with quadratic complexity in the model of one-time-only programs (cf. Remarks 5.2 and 5.3).

For all $k \in \mathbb{N}$, $(k, *)$-programs of bounded width are not more powerful than $(1, *)$-programs (cf. Remark 5.4). This fact is not true for general k-times-only programs (cf. Remark 5.5).

Nondeterministic branching programs of bounded width and polynomial size which are one-time-only programs with regard to the nondeterministic variables compute exactly those languages of NC^1 [B; M]. However, all languages in NP can be computed by polynomial size nondeterministic branching programs of bounded width which are one-time-only programs with regard to the deterministic and $(2, *)$-programs with regard to the nondeterministic variables [M].

Using a theorem of MEHLHORN–SCHMIDT our technique leads

immediately to lower bounds on the communication complexity COMM(f) of Boolean functions f. This complexity measure was firstly discussed by Yao [Y] and corresponds to the amount of information transfer necessary to compute the function.

Lower bounds on communication complexity provide lower bounds on VLSI-complexity. In particular, if A is the area and T the time of a where-oblivious VLSI-circuit computing the function f then $AT^2 \geqslant \Omega((\text{COMM}(f))^2)$ (see, e.g., [Y]). The best lower bound on communication complexity is an $\Omega(n^2)$ bound for the decision whether two graphs over n nodes are isomorphic [Y]. In Section 6 we prove a sharp bound of order $\Theta(n \cdot \log(n))$ on $\text{COMM}(\text{POLY}^*_{n,\delta n})$.

In Section 2 basic definitions and denotations will be given.

2. Basic Definitions and Remarks

For any given positive $n \in \mathbb{N}$, let X_n denote a collection of n Boolean variables $\{x_1, ..., x_n\}$. B_n stands for the set of Boolean functions over X_n, i.e., $B_n := \{f: \{0,1\}^n \to \{0,1\}\}$.

A branching program P over X_n is an acyclic, directed, and labelled graph with a single root (a node of indegree 0) and two sinks (nodes of outdegree 0). Each non-sink of P has outdegree 2 and is labelled by a variable from X_n. Edges and sinks are labelled by constants from $\{0,1\}$, where edges leaving the same non-sink have distinct labels.

Let P be a branching program over X_n. Each input vector $w \in \{0,1\}^n$ corresponds to exactly one directed path p_w in P leading from the root to one of the sinks. The path p_w starts in the root. If it reaches a non-sink with label $x_i \in X_n$, $1 \leqslant i \leqslant n$, then p_w follows the edge with label $w_i \in \{0,1\}$.

In this way each branching program over X_n computes exactly one Boolean function $f \in B_n$. The value $f(w)$ is given by the label of the sink reached by p_w. For any branching program P we denote the number of non-sinks in P by size(P) and the length of a longest directed path in P by depth(P).

The set of languages that can be computed by sequences of polynomial size branching programs is equal to the complexity class LOGSPACE(nonuniform) ($\mathcal{L}$) [PZ]. Thus, lower bound techniques for branching programs can be used for the separation of L from NL, P, or NP.

For given inputs w, $w' \in \{0,1\}^n$ we write $w \leqslant w'$ iff for all i, $1 \leqslant i \leqslant n$, $w_i \leqslant w'_i$. For a given subset Y of X_n we denote the set of assignments of Y by 2^Y, i.e., $2^Y = \{c: Y \to \{0,1\}\}$. Each pair (c, c'), $c \in 2^Y$, $c' \in 2^{X_n \setminus Y}$, of "disjoint" assignments defines exactly one input word $w = c \cup c' \in \{0,1\}^n$. Conversely, for any $w \in \{0,1\}^n$, we denote the assignment of Y which coincides

with w on Y by $w|_Y$, i.e., $w = w|_Y \cup w|_{X_n \setminus Y}$. Let f be a Boolean function over X_n, Y a subset of X_n, and c an assignment of Y. The subfunction f^c of f over the set $X_n \setminus Y$ of Boolean variables is defined as follows:

Let $f^c(c') = f(c \cup c')$ for all assignments c' of $X_n \setminus Y$. Let P be a branching program over X_n. Let $p = ((v_1, b_1), ..., (v_q, b_q))$, $q \in \mathbb{N}$, be a path in P containing the non-sinks $v_1, ..., v_q$ and leading from v_1 to the node v_{q+1}, where for all i, $1 \leqslant i \leqslant q$, $b_i \in \{0, 1\}$ denotes the label of the edge (v_i, v_{i+1}). We call p consistent with the assignment c of Y if for all i, $1 \leqslant i \leqslant q$, the label x of v_i is in Y and $c(x) = b_i$.

3. The Lower Bound Technique

Let us first formulate what we understand by the "alternation depth" of a given branching program P over X_n with regard to a fixed subset Y of X_n.

Let $p = ((v_1, b_1), ..., (v_q, b_q))$, $q \in \mathbb{N}$, be a path in P leading from the non-sink v_1 to the node v_{q+1} of P. We write $p = p_1 \circ p_2$ if there is an index r, $1 \leqslant r \leqslant q$, so that $p_1 = ((v_1, b_1), ..., (v_r, b_r))$ and $p_2 = ((v_{r+1}, b_{r+1}), ..., (v_q, b_q))$. Let $Y \subseteq X_n$ and p' be a path in P. We define p' to be Y-polarized if there is a decomposition $p' = p'_1 \circ p'_2$ of p', so that p'_1 contains only non-sinks with labels from Y and p'_2 does not contain non-sinks with labels from Y, or vice versa. We call p' strongly Y-polarized if $p'_2 \neq 0$. If a path p' in P is Y-polarized with the corresponding decomposition $p' = p'_1 \circ p'_2$ then we denote the first node of p'_1 by $\mathrm{beg}(p')$. If p' is strongly Y-polarized $\mathrm{mid}(p')$ denotes the first node of p'_2.

For a path p in P we denote by $\mathrm{altdepth}(p, Y)$ the length of a shortest decomposition of p into Y-polarized subpaths. Let $\mathrm{altdepth}(P, Y) = \max\{\mathrm{altdepth}(p, Y); p \text{ is path in } P\}$. Let f be a Boolean function over X_n and Y a subset of X_n. We call a set $C \subseteq \{0, 1\}^n$ of inputs a (f, Y)-bundle if for all distinct $w, w' \in C$,

(1) $w|_Y \neq w'|_Y$ and $w|_{X_n \setminus Y} \neq w'|_{X_n \setminus Y}$ and

(2) $f(w) = f(w') = 1$, but $f(w|_Y \cup w'|_{X_n \setminus Y}) = f(w'|_Y \cup w|_{X_n \setminus Y}) = 0$.

Let $\beta(f, Y) = \max\{|C|; C \subseteq \{0, 1\}^n \text{ is } (f, Y)\text{-bundle}\}$.

Our results are based on the following

THEOREM 1. *Let P be a branching program over X_n computing $f \in B_n$. Further let $Y \subseteq X_n$, $k = \mathrm{altdepth}(P, Y)$ and C be a (f, Y)-bundle. Then* $\mathrm{size}(P) \geqslant |C|^{1/(2k-1)}$.

Proof. For all $w \in \{0, 1\}^n$ let $l_w = \mathrm{altdepth}(p_w, Y)$, and $p_w = p_{w,1} \circ \cdots \circ p_{w,l_w}$ be the decomposition of p_w into Y-polarized subpaths. Observe that $l_w \leqslant k$, and that $p_{w,1}, ..., p_{w,l_w-1}$ are strongly Y-polarized.

For any $w \in \{0,1\}^n$ the vector $V_w = (v_{w,1}, u_{w,1}, ..., v_{w,k}, u_{w,k})$, consisting of nodes from P, is defined as follows.

For all j, $1 \leqslant j \leqslant l_w - 1$, let $v_{w,j} = \text{beg}(p_{j,w})$ and $u_{w,j} = \text{mid}(p_{j,w})$. Let $v_{w,l_w} = \text{beg}(p_{w,l_w})$. If p_{w,l_w} is strongly Y-polarized let $u_{w,l_w} = \text{mid}(p_{w,l_w})$, else let u_{w,l_w} be the sink which p_w is leading to.

For all j, $l_w < j \leqslant k$, let $v_{w,j}$ and $u_{w,j}$ be the sink which p_w is leading to. Observe that $v_{w,1}$ is always the root of P. Clearly, for all w, $w' \in \{0,1\}^n$ from $V_w = V_{w'}$ it follows that $V_{w_1} = V_{w_2} = V_w = V_{w'}$, where $w_1 = w|_Y \cup w'|_{X_n \setminus Y}$ and $w_2 = w'|_Y \cup w|_{X_n \setminus Y}$. By the definition of a (f, Y)-bundle we obtain that for all $w \neq w' \in C$ it holds that $V_w \neq V_{w'}$. Hence, $|C| \leqslant |\{V_w; w \in \{0,1\}^n\}| \leqslant \text{size}(P)^{2k-1}$. ∎

Let us define the following measures for branching programs P over X_n and Boolean functions $f \in B_n$. For all m, $1 \leqslant m < n$, let $\text{altdepth}(P, m) = \min\{\text{altdepth}(P, Y);\ Y \subseteq X_n \text{ and } |Y| = m\}$ and $\beta(f, m) = \min\{\beta(f, Y); Y \subseteq X_n \text{ and } |Y| = m\}$.

From Theorem 1 we obtain directly the following

COROLLARY 3.1. *Let P be a branching program over X_n computing $f \in B_n$. Then for all m, $1 \leqslant m < n$, and $k = \text{altdepth}(P, m)$ we have* $\text{size}(P) \geqslant \beta(f, m)^{1/(2k-1)}$.

4. APPLICATIONS

In this section for three explicitly defined sequences $(f_n)_{n \in \mathbb{N}}$ of Boolean fucntions we prove lower bounds on $\beta(f_n, m)$ which are exponential in n if $m = \Omega(n)$. But first let us give some technical definitions about Boolean matrices.

For positive natural numbers t, n let $X_{t,n}$ denote a collection $X_{t,n} = \{x_{1,1}, ..., x_{1,n}, ..., x_{t,1}, ..., x_{t,n}\}$ of Boolean variables. Let $M_{t,n}$ denote the set of assignments of $X_{t,n}$, i.e., the set of Boolean matrices with t rows and n columns.

For a given matrix $A \in M_{t,n}$, a subset $Y \subseteq X_{t,n}$, and a number j, $1 \leqslant j \leqslant n$, let $|A|_j$ denote the number of 1's in the jth column of A, and let $|Y|_j$ denote the number of indices i, $1 \leqslant i \leqslant n$, with $x_{i,j} \in Y$.

By $w(A) \in \{0, ..., t\}^n$ we denote the string of column sums of A, i.e., $w(A) = (|A|_1, ..., |A|_n)$.

In general for a word $w \in \{0, ..., t\}^*$, $t \geqslant 1$, let $l(w)$ denote the number of components of w belonging to $\{0,1\}$, and let $w^* \in \{0,1\}^{l(w)}$ be the word obtained from w by deleting all components of w which are not in $\{0,1\}$. For example, for $w = (2013202130) \in \{0,1,2,3\}^{10}$ we get $l(w) = 5$ and $w^* = (01010)$.

We define the language $\mathrm{SET} = (\mathrm{SET}_n)_{n \in \mathbb{N}}$. For all $n \in \mathbb{N}$ and all A, $A' \in M_{2,n}$ let $\mathrm{SET}_n(A, A') = 1 \Leftrightarrow w(A)^* = w(A')^*$. Obviously, SET_n can be thought to be a Boolean function over $X_{2,2n}$.

LEMMA 4.1. *For all natural n, $k \in \mathbb{N}$, $0 < k < 4n$, it holds that $\beta(\mathrm{SET}_n, k) \geqslant 2^{k'/4}$, where $k' = \min\{k, 4n-k\}$.*

Proof. As for all Boolean functions f over X_n and all $Y \subseteq X_n$, $\beta(f, X_n \setminus Y) = \beta(f, Y)$ we can w.l.o.g. suppose that $k \leqslant 2n = |X_{2,2n}|/2$, i.e., $k = k'$.

Let Y be a subset of $X_{2,2n}$. We have to construct a (SET_n, Y)-bundle of cardinality $2^{k/4}$. Let $X_{2,2n}$ be partitioned into X' and X'', where $X' = \{x_{1,1}, x_{2,1}, ..., x_{1,n}, x_{2,n}\}$ and $X'' = \{x_{1,n+1}, x_{2,n+1}, ..., x_{1,2n}, x_{2,2n}\}$.

We suppose that $|Y \cap X'| \supseteq |Y \cap X''|$. Obviously, not less than $k/4$ columns of X' contain elements of Y, and, as $k \leqslant 2n$, not less than $k/4$ columns of X'' contain elements of $X_{2,2n} \setminus Y$. For $p = k/4$ we fix indices $i_1 < \cdots < i_p$ and $j_1 < \cdots < j_p$ from $\{1, ..., n\}$ and vectors $(a_1, ..., a_p)$ and $(b_1, ..., b_p)$ over $\{1, 2\}$ so that for all numbers q, $1 \leqslant q \leqslant p$, $x_{a_q, i_q} \in Y$ and $x_{b_q, n+j_q} \in X_{2,2n}/Y$.

Let $K = \{(a_1, i_1), ..., (a_p, i_p), (b_1, n+j_1), ..., (b_p, n+j_p)\}$. To each $w \in \{0,1\}^p$ we assign a $2 \times 2n$ matrix A^w. If for natural i, j, $1 \leqslant i \leqslant 2$, $1 \leqslant j \leqslant 2n$, (i, j) is not in K, let $A^w_{i,j} = 0$. If there is a number q, $1 \leqslant q \leqslant p$, so that $(i, j) = (a_q, i_q)$ or $(i, j) = (b_q, n+j_q)$, let $A^w_{i,j} = w_q$.

It is easy to check that for all w, $w' \in \{0,1\}^p$ and $A = A^w|_Y \cup A^{w'}|_{X_{2,2n}} = A^w|_{X'} \cup A^{w'}|_{X''}$ it holds that $w(A)^* = ww'$. Hence, $C = \{A^w : w \in \{0,1\}^p\}$ is a (SET_n, Y)-bundle. ∎

Let $n \in \mathbb{N}$ be a prime number. We denote by $\mathbb{F}_n$ the corresponding field $\{0, ..., n-1\}$ and suppose that rows and columns of matrices from $M_{n,n}$ are numbered with elements from $\mathbb{F}_n$. For each mapping $g: \mathbb{F}_n \to \mathbb{F}_n$ we denote by $A^g \in M_{n,n}$ the following matrix. For all i, $j \in \{1, ..., n\}$ let $A^g_{i,j} = 1 \Leftrightarrow j = g(i)$. For all numbers $d < n$ we define the decision problem $\mathrm{POLY}^*_{n,d}$ as follows.

For all matrices $A \in M_{n,n}$ let $\mathrm{POLY}^*_{n,d}(A) = 1 \Leftrightarrow$ there is a polynomial p: $\mathbb{F}_n \to \mathbb{F}_n$ of degree not greater than d so that $A^p = A$. Obviously, $\mathrm{POLY}^*_{n,d}$ is a Boolean function over the set $X_{n,n} = \{x_{i,j};\ i, j \in \mathbb{F}_n\}$ of Boolean variables. We prove the following lower bound on β.

LEMMA 4.2. *Let d and k be natural numbers fixed in such a way that d is odd, $d+1 \leqslant n/2$ and $k \leqslant n^2/2$. Then $\beta(\mathrm{POLY}^*_{n,d}, k) \geqslant (\min\{n^{\lfloor k/n \rfloor}, n^{(d+1)/2}, \lfloor k/n \rfloor^{(d+1)/2}\})/2^{n-d-1}$.*

Proof. We denote by $\mathcal{POLY}(n, d)$ the set of polynomials over $\mathbb{F}_n$ of degree not greater than d. We make use of the following property of polynomials over a field.

(I) Let $(a_1, ..., a_{d+1}), (b_1, ..., b_{d+1}) \in \mathbb{F}_n^{d+1}$ be arbitrarily fixed. Then there is exactly one polynomial $p \in \mathcal{POLY}(n, d)$ so that $p(a_i) = b_i$ for all i, $1 \leqslant i \leqslant d+1$.

For subsets I of $\mathbb{F}_n$ and Y and $X_{n,n}$ we denote by $\mathrm{MAP}(Y, I)$ the set of all mappings $m: I \to \mathbb{F}_n$ with the property that for all $i \in I$, $x_{i,m(i)} \in Y$ if $|Y|_i > 0$ and $m(i) = 0$ if $|Y|_i = 0$. Obviously, $|\mathrm{MAP}(Y, I)| = \prod_{i \in I} (\max\{(Y|_i, 1\})$.

Let Y be a subset of $X_{n,n}$ with k elements and $Z = X_{n,n} \setminus Y$. We fix an numbering $\{i_1, ..., i_n\}$ of $\mathbb{F}_n$ so that for all q, $1 \leqslant q \leqslant n-1$, $|Y|_{i_q} \geqslant |Y|_{i_{q+1}}$.

Let $e = (d+1)/2$, $I = \{i_1, ..., i_e\}$, $J = \{i_{n-e+1}, ..., i_n\}$, and $K = \{1, ..., n\} \setminus (I \cup J)$. As $e < n/2$ we get $I \cap J = 0$ and $|I \cup J| = d+1$.

Let $a = a(Y) = \min\{|\mathrm{MAP}(Y, I)|, |\mathrm{MAP}(Z, J)|\}$, and let us fix pairwise distinct mappings $r_1, ..., r_a$ in $\mathrm{MAP}(Y, I)$ and $s_1, ..., s_a$ in $\mathrm{MAP}(Z, J)$.

Due to (I) for all i, $1 \leqslant i \leqslant a$, there is exactly one polynomial $p = p_i$ in $\mathcal{POLY}(n, d)$ with the property that $p|_I = r_i$ and $p|_J = s_i$. Let us denote by Π the set $\Pi = \{p_1, ..., p_a\}$. For all $p \in \Pi$ we denote by K_p the set $K_p = \{k \in K;\ x_{k,p(k)} \in Y\}$. According to the pigeon hole principle there is a subset Π^* of Π of cardinality at least $a(Y)/2^{|K|}$ with the property that for all p, $p' \in \Pi^*$ it holds that $K_p = K_{p'}$.

We claim that $C = \{A^p;\ p \in \Pi^*\}$ is a $(\mathrm{POLY}^*_{n,d}, Y)$-bundle. Indeed, by the definition of Π for all $p \neq p' \in \Pi^*$ we have $A^p|_Y \neq A^{p'}|_Y$ and $A^p|_Z \neq A^{p'}|_Z$.

Let $p \neq p' \in \Pi^*$ be arbitrarily fixed, $A = A^p|_Y \cup A^{p'}|_Z$ and $B = A^{p'}|_Y \cup A^p|_Z$. It is to show that $\mathrm{POLY}^*_{n,d}(A) = \mathrm{POLY}^*_{n,d}(B) = 0$.

As $d+1 \leqslant n/2$ it holds that $|K| \geqslant d+1$. Let us suppose that $|K_p| \geqslant |K \setminus K_p|$. (Otherwise take Z instead of Y.) If there exists a polynomial $p'' \in \mathcal{POLY}(n, d)$ with $A^{p''} = A$ or $A^{p''} = B$ then p'' coincides with p on $K_p \cup I$ or with p' on $K_p \cup I$, respectively. As $|K_p \cup I| \geqslant e + (d+1)/2 = 2e = d+1$, we obtain $p'' = p$ or $p'' = p'$, respectively. But this contradicts the definition of Π.

It remains to estimate $a(Y)$. It is easy to verify that $a(Y)$ has a "local minimum" if

(1) Y contains the set $\{x_{i,j};\ 0 \leqslant i \leqslant \lfloor k/n \rfloor - 1,\ 0 \leqslant j \leqslant n-1\}$ or

(2) Y contains the set $\{x_{i,j};\ 0 \leqslant i \leqslant n-1,\ 0 \leqslant j \leqslant \lfloor k/n \rfloor - 1\}$.

Because of $k \leqslant n^2/2$ we obtain from case (1) that $a(Y) = \lfloor k/n \rfloor^e$ and from case (2) that $a(Y) = \min\{n^{\lfloor k/n \rfloor}, n^e\}$. ∎

The proof of Lemma 4.2 is also valid for the corresponding monotone problem $\mathrm{POLY}_{n,d}$ over $X_{n,n}$ which is defined as follows: For all $n \in \mathbb{N}$ and $A \in M_{n,n}$ let $\mathrm{POLY}_{n,d}(A) = 1 \Leftrightarrow \exists p \in \mathcal{POLY}(n, d)$ with $A^p \leqslant A$. For all $\delta \in (0, 1)$ the problem $\mathrm{POLY}_\delta = (\mathrm{POLY}_{n,\delta n})$ is NP-complete (see, e.g., [A1]).

5. Some Remarks about k-Times-Only Branching Programs

For all $k \in \mathbb{N}$ we call a branching program P a k-times-only program if each path in P contains not more than k non-sinks with the same label.

One-time-only programs are just those branching programs without null chains. They were investigated by many authors. The first exponential lower bounds on the complexity of one-time-only branching programs were proved in [PZ; W] for clique functions over undirected graphs. Further exponential lower bounds for a big number of algebraic and graph-theoretic problems can be found in [J; A&; D; KW; K1].

There are languages computable only by exponential size one-time-only programs which can be computed by polynomial size 2-times-only programs. The very first example of a language of this kind was given in [W]. But today we have no method of proving lower bounds for k-times-only programs ($k \geqslant 2$) which is better than that for the general model. This is the motivation for the consideration of the following restricted model.

A program P is called levelled if its nodes are organized in disjoint levels, where for each edge (u, v) from P u is in level i and v is in level $i+1$ for some i, $i < \text{depth}(P)$. For any levelled branching program P let $\text{width}(P)$ denote the maximum taken over the cardinalities of all levels of P.

We call P an oblivious branching program if P is levelled and all non-sinks on the same level have the same label. Let P be an oblivious one-time-only branching program over the set $X_n = \{x_1, ..., x_n\}$ of Boolean variables. We suppose, w.l.o.g., that P is of depth n. Obviously, P is characterized by a permutation $\sigma \in \mathbb{S}_n$ in the sense that for all m, $1 \leqslant m \leqslant n$, non-sinks at level m have label $x_{\sigma(m)}$.

We investigate the following restricted model. For all $\sigma \in \mathbb{S}_n$ and $k \in \mathbb{N}$, an oblivious k-times-only program P shall be called (k, σ)-program if $\text{depth}(P) = kn$, and if for all m, j, $1 \leqslant m \leqslant n$, $0 \leqslant j \leqslant k-1$, non-sinks at level $jn+m$ have label $x_{\sigma(m)}$.

We call an oblivious k-times-only branching program P over X_n $(k, *)$-program if $\exists \sigma \in \mathbb{S}_n$ so that P is (k, σ)-program. Thus, the model of $(k, *)$-programs is a natural generalization of oblivious one-time-only branching programs and consists of such oblivious programs which are allowed to read the input word k times in succession, but each time in the same order. From our considerations in Section 3 we obtain a method for proving exponential lower bounds on the complexity of $(k, *)$-programs for all $k \in \mathbb{N}$.

Lemma 5.1. *Let $k, n \in \mathbb{N}$ and P be a $(k, *)$-program over X_n. Then for all m, $1 \leqslant m \leqslant n-1$, we have* $\text{altdepth}(P, m) \leqslant k$.

Proof. We fix $\sigma \in \mathbb{S}_n$ so that P is a (k, σ)-program. For all m,

$1 \leqslant m \leqslant n-1$, let $Y_m = \{x_{\sigma(1),\ldots,\sigma(m)}\}$. By definition we obtain that altdepth$(P, Y_m) = k$ for all m, $1 \leqslant m < n$. ▮

For each Boolean function f over X_n we define $\beta(f)$ as to be $\beta(f) = \max\{\beta(f, m); 1 \leqslant m < n\}$. From Theorem 1 we obtain

COROLLARY 5.2. *Let f be a Boolean function over X_n, $k \in \mathbb{N}$, and let P be a $(k, *)$-program over X_n computing f. Then* size$(P) \geqslant \beta(f)^{1/(2k-1)}$. ▮

Thus, Lemmas 4.1 and 4.2 provide lower bounds on the complexity of $(k, *)$-programs of order $\exp(\Omega(n))$ for the problem SET, and of order $\exp(\Omega(n^{1/2}\log(n)))$ for the problems POLY_δ^* and POLY_δ, $\delta \in (0, \frac{1}{2})$.

In the following remarks we relate the computational power of $(k, *)$-programs to that of unrestricted oblivious k-times-only programs and unrestricted one-time-only programs ($k \in \mathbb{N}$). Therefore let us denote by $\mathscr{P}_{BPk}$, $\mathscr{P}_{o\text{-}BPk}$, $\mathscr{P}_{(k,*)}$ the classes of languages computable by polynomial size k-times-only-, oblivious k-times-only-, $(k, *)$-programs, respectively.

Remark 5.3. For all $k \geqslant 2$, $\mathscr{P}_{(k,*)}$ is not contained in $\mathscr{P}_{BP1}$.

Sketch of the Proof. We define the language $\text{POSITION} = (\text{POS}_n)_{n \in \mathbb{N}}$, where POS_n is a Boolean function over $X_{3,n}$.

For each matrix $A \in M_{3,n}$ let $2(A)$, $3(A)$ denote the number of letters 2, 3, resp. in $w(A)$, i.e., the number of columns in A with column sum 2, 3, respectively. Observe $2(A) + 3(A) + l(w(A)) = n$. Let $\text{POS}_n(A) = 1$ iff $1 \leqslant 2(A) \leqslant l(w(A))$ and the $2(A)$th component of $w(A)^*$ is one or $1 \leqslant 3(A) \leqslant l(w(A))$ and the $(l(w(A)) - 3(A) + 1)$th component of $w(A)^*$ is one (see Section 4).

In [K1] it is shown that all sequences of one-time-only programs computing the language POSITION have size $\exp(\Omega(n))$. We show that for all $n \in \mathbb{N}$ the function POS_n can be computed by $(2, *)$-programs of size $O(n^3)$. POS_n can be written as $PT_n = OR_{1 \leqslant l, a \leqslant n}(h_{l,a}^n \wedge H_{l,a}^n)$ (I), where for all $A \in M_{3,n}$, $h_{l,a}^n(A) = 1 \Leftrightarrow l(w(A)) = l$ and $2(A) = a$, and $H_{l,a}^n(A) = 1 \Leftrightarrow 1 \leqslant a \leqslant l$ and $w(A)_a^* = 1$ or $1 \leqslant n - (l + a) \leqslant l$ and $w(A)_{l+a+1}^* = 1$.

It is not difficult to prove that both $h_{l,a}^n$ and $H_{l,a}^n$ can be computed by $(1, \sigma_n)$-programs of quadratic size in n, where σ_n: $\{1, \ldots, 3n\} \to \{1, 2, 3\} \times \{1, \ldots, n\}$ is defined as $\sigma_n(q + 3p) = (q, p + 1)$ for all integer p, q, $0 \leqslant p \leqslant n-1$, $1 \leqslant q \leqslant 3$.

Thus, according to (I) we can construct a $(2, \sigma_n)$-program of size $O(n^3)$ which computes POS_n. ▮

Hence, as in the general case, $\mathscr{P}_{(1,*)}$ is a proper subset of $\mathscr{P}_{(k,*)}$ for all $k \geqslant 2$. The following two remarks show that neither polynomial size unrestricted one-time-only programs nor polynomial size oblivious k-times-

only programs can be simulated by polynomial size $(K, *)$-programs for some $K \in \mathbb{N}$.

Remark 5.4. For all $k \in \mathbb{N}$ the class $\mathcal{P}_{BP1}$ is not contained in $\mathcal{P}_{(k,*)}$.

Proof. In Lemma 4.1 we have shown that the language SET is not in $\mathcal{P}_{(k,*)}$ for all $k \in \mathbb{N}$. But SET can be computed by one-time-only programs of quadratic size [K2]. ∎

Remark 5.5. For all $k \in \mathbb{N}$ the class $\mathcal{P}_{o\text{-}BP2}$ is not contained in $\mathcal{P}_{(k,*)}$.

Sketch of the Proof. Let us consider the permutation matrix problem $\mathrm{PMP} = (\mathrm{PMP}_n)_{n \in \mathbb{N}}$, where PMP_n is the following Boolean function over $X_{n,n}$. For all $A \in M_{n,n}$ let $\mathrm{PMP}_n(A) = 1 \Leftrightarrow A$ is a permutation matrix, i.e., $\exists \sigma \in \mathbb{S}_n$ so that $A = A^\sigma$.

By similar arguments as in the proof of Lemma 4.2 it is possible to show that $\beta(\mathrm{PMP}_n) = \exp(\Omega(n))$ [K2], i.e., for all $k \in \mathbb{N}$ the permutation matrix problem PMP is not in $\mathcal{P}_{(k,*)}$. On the other hand, a matrix $A \in M_{n,n}$ is a permutation matrix if and only if each row and each column of A contains exactly one "1." The test E_n^1 whether a given vector $v \in \{0,1\}^n$ contains exactly one "1" can be performed by oblivious one-time-only programs of width 3.

Thus, we can construct an oblivious 2-times-only program of width 3 over $X_{n,n}$ which computes PMP_n by verifying E_n^1 for each row and each column. ∎

Let us make some remarks about oblivious k-times-only programs of bounded width. We denote by $\mathcal{P}_{bw\text{-}o\text{-}BPk}$, $\mathcal{P}_{bw\text{-}(k,*)}$ the set of languages computable by sequences of oblivious k-times-only-, $(k, *)$-programs, respectively, of bounded width.

In [J] it is proved that the permutation matrix problem does not belong to $\mathcal{P}_{BP1}$. By Remark 5.5 we obtain that $\mathcal{P}_{bw\text{-}o\text{-}BP1}$ is a proper subset of $\mathcal{P}_{bw\text{-}o\text{-}BP2}$. This does not hold for $(k, *)$-programs. We prove the following

Remark 5.6. For all $k \in \mathbb{N}$ it holds that $\mathcal{P}_{bw-(k,*)} = \mathcal{P}_{bw\text{-}(1,*)}$.

Sketch of the Proof. Let $k, n \in \mathbb{N}$ and $\sigma \in \mathbb{S}_n$ be arbitrarily fixed, and let P be a (k, σ)-program over X_n of width d computing the Boolean function f. We prove our claim by showing that there is a $(1, \sigma)$-program Q of width $(d+1)^{dk}$ which computes f. For each m, $1 \leqslant m < n$, let $Y_m = \{x_{\sigma(1)}, \ldots, x_{\sigma(m)}\}$ and $y_m = |\{f^c;\ c \text{ assignment of } Y_m\}|$.

It is not difficult to prove that there is a $(1, \sigma)$-program Q which computes f and in which each level m, $1 \leqslant m \leqslant n-1$, contains exactly y_m nodes [K1].

We have to estimate $\max\{y_m;\ 1 \leqslant m \leqslant n-1\}$. Suppose that for each j, $1 \leqslant j \leqslant kn$, level j of P contains $d(j)$ nodes numbered by $(1, j), \ldots, (d(j), j)$.

Let m, $1 \leqslant m \leqslant n-1$, and $c \in 2^Y m$ be arbitrarily fixed. Remember that for all r, $0 \leqslant r \leqslant k-1$, the nodes at levels $rn+1, ..., rn+m$ are labelled with variables from Y_m. Thus, for all natural r, $0 \leqslant r \leqslant k-1$, and q, $1 \leqslant q \leqslant d(rn+1)$, there is exactly one path $p_{c,r,q}$ in P which is consistent with c and which starts at the qth node of level $rn+1$, i.e., the node numbered by $(q, rn+1)$.

Clearly, $p_{c,r,q}$ leads to some node v on level $rn+m+1$. Let $\phi(c, r, q)$ denote the number of v, i.e., v has the number $(\phi(c, r, q), rn+m+1)$. Thus, each assignment c of Y_m corresponds to exactly one vector $V(c) \in \{0, ..., d\}^{kd}$ which is defined as follows.

Let s, $1 \leqslant s \leqslant kd$, be arbitrarily fixed, $s = rn+q$, $1 \leqslant r < k$, $1 \leqslant q \leqslant d$. Let $V(c)_s = \phi(c, r, q)$ if $q \leqslant d(rn+1)$, and $V(c)_s = 0$ if not. It is not hard to show that for all assignments c, c' of Y_m with $V(c) = V(c')$ it holds $f^c = f^{c'}$. ∎

Let us conclude this section with some remarks about nondeterministic branching programs. For n, $m \in \mathbb{N}$ let $X_n = \{x_1, ..., x_n\}$ and $Y_m = \{y_1, ..., y_m\}$ be two disjoint subsets of Boolean variables and P a branching program over $X_n \cup Y_m$. By modifying the definition of acceptance, P can be considered to be a nondeterministic branching program over X_n. We say that P accepts an assignment c of X_n if there is an assignment c' of Y_m so that P computes 1 on $c \cup c'$.

Using Barrington's result [B] that $\mathscr{P}_{bw\text{-}BP} = \mathscr{NC}^1$, in [M] it is shown that the set of languages computable by polynomial size nondeterministic branching programs of bounded width which are one-time-only programs with regard to the nondeterministic variables equals $\mathscr{NC}^1$, too.

But all languages in $\mathscr{NP}$ can be computed by sequences of polynomial size oblivious nondeterministic branching programs of width 3 which are $(1, *)$-programs with regard to the deterministic and $(2, *)$-programs with regard to the nondeterministic variables [M].

6. Bounds on Communication Complexity

Following [Y; MS] the communication complexity of a Boolean function f over X_n is defined as follows.

Let (Y, Z) be a balanced partition of X_n, i.e., $Y \cap Z = 0$, $Y \cup Z = X_n$, and $||Y| - |Z|| \leqslant 1$. Let L and R be two processors which know only the input variables of Y and Z, respectively. In order to compute $f(v)$ for given $v \in \{0, 1\}^n$ they send information to each other alternately, one bit at a time, according to some algorithm $\mathscr{A}$. The amount of information transfer necessary to compute $f(v)$ is determined by the number of bits which have to be exchanged. More exactly, a communication algorithm $\mathscr{A}$ with respect

to (Y, Z) is given by two response functions $h_L: 2^Y \times \{0,1\}^* \to \{0,1\}$ and $h_R: 2^Z \times \{0,1\}^* \to \{0,1\}$ and a partial output function $a: \{0,1\} \to \{0,1\}$.

Processor L starts the computation and sends $w_1 = h_L(v')$ to processor R (let $v' = v|_Y$ and $v'' = v|_Z$). Processor R returns $w_2 = h_R(v'', w_1)$, L returns $w_3 = h_L(v', w_1 w_2)$, ... until $w_1 w_2 \cdots w_k$ belongs to $\mathrm{dom}(a)$. At this point the computation stops with the result $f(v) = a(w_1, .., w_k)$, where $k = k_{\mathscr{A}}(v)$ denotes the amount of information transfer required by $\mathscr{A}$ on input v.

By $k_{\mathscr{A}}$ we denote the value $k_{\mathscr{A}} = \max\{k_{\mathscr{A}}(v),\ v \in \{0,1\}^n\}$ and by $\mathrm{COMM}(f, Y, Z)$, the minimum over all $k_{\mathscr{A}}$, where $\mathscr{A}$ is a communication algorithm computing f with respect to (Y, Z). The communication complexity $\mathrm{COMM}(f)$ of f is defined as $\mathrm{COMM}(f) = \min\{\mathrm{COMM}(f, Y, Z);$ (Y, Z) is a balanced partition of $X_n\}$.

We prove the following

THEOREM 6.1. $\mathrm{COMM}(f) \geqslant \log_2 \beta(f, \lfloor n/2 \rfloor)$ *for all* $f \in B_n$.

Proof. Let $\mathscr{A}$ be the communication algorithm which provides $k_{\mathscr{A}} = \mathrm{COMM}(f)$, where $\mathscr{A}$ is defined with respect to some balanced partition (Y, Z) of X_n. The proof is an immediate consequence of a theorem of Mehlhorn and Schmidt [MS] stating $\mathrm{COMM}(f) \geqslant \log_2(\mathrm{rank}(M^f_{(Y,Z)})$, where $M^f_{(Y,Z)} = (f(v' \cup v''))_{v' \in 2^Y, v'' \in 2^Z}$ denotes the communication matrix of f with respect to (Y, Z).

It is easy to see that for each (f, Y)-bundle C, $C \subseteq \{0,1\}^n$, the $|C| \times |C|$ submatrix of $M^f_{(Y,Z)}$, induced by all inputs from C, has rank $|C|$. ∎

Obviously, for each $f \in B_n$ we have $\mathrm{Comm}(f) = O(n)$. Hence, Lemma 4.1 yields

THEOREM 6.2. $\mathrm{COMM}(\mathrm{SET}_n) = \Theta(n)$.

Moreover, it is easy to see that for each balanced partition (Y, Z) of $X_{n,n}$ ($n \in \mathbb{N}$ prime), and any d, $d \leqslant n$, the function $\mathrm{POLY}^*_{n,d}$ can be calculated by a communication algorithm $\mathscr{A}$ with respect to (Y, Z) with $k_{\mathscr{A}} = O(n \cdot \log(n))$. (On each matrix $A \in M_{n,n}$ the processors L and R simultaniously produce the word $\mathrm{bin}(0) \# \mathrm{bin}(g(0)) \# (\mathrm{bin}(1) \# \mathrm{bin}(g(1)) \# \cdots \# \mathrm{bin}(n-1) \# \mathrm{bin}(g(n-1))$ if A equals the graph A^g for some mapping $g: \mathbb{F}_n \to \mathbb{F}_n$ and some "bad" word if not. The partial output function rejects all "bad" words or decides whether g is a polynomial of degree not greater than d.) According to Lemma 4.2 we obtain

THEOREM 6.3. $\mathrm{COMM}(\mathrm{POLY}^*_{n,\delta n}) = \Theta(n \cdot \log(n))$ *for all* $\delta \in (0, \frac{1}{2})$.

ACKNOWLEDGMENTS

I thank Oleg B. Lupanov from Moscow, Stasys Jukna from Vilnius, and Christoph Meinel and Stefan Waack from Berlin for many helpful discussions.

RECEIVED January 4, 1988; FINAL MANUSCRIPT RECEIVED August 17, 1989

REFERENCES

[A1] ANDREEV, A. E. (1985), On a method of obtaining lower bounds to the complexity of individual monotone functions, *Dokl. Akad. Nauk SSSR* **282**, No. 5, 1033–1037.

[A2] ANDREEV, A. E. (1986), A method of obtaining superquadratic lower bounds on the complexity of Π-schemes. *Vestnik. Moscow Ună. Ser. I Mat. Mech.* **6**, 73–75.

[AB] ALON, N., AND BOPPANA, R. B. (1987), The monotone circuit complexity of Boolean functions, *J. Combinatorika* **7**, No. 1, 1–22.

[A&] AJTAI, M., BABAJ, L., HAJNAL, P., KOMLOS, J., PUDLAK, P., RÖDEL, V., SZEMEREDI, E., AND TURAN, G. (1986), Two lower bounds for branching programs, *in* "Proceedings ACM STOC", pp. 30–39.

[B] BARRINGTON, D. A. (1986), Bounded width-polynomial size branching programs recognize exactly those languages in NC^1, *in* "Proceedings 18th ACM STOC," pp. 1–5.

[Bl] BLUM, N. (1984), A Boolean function requiring $3n$ network size, *J. Theoret. Comput. Sci.* **28**, 337–345.

[D] DUNNE, P. E. (1985), Lower bounds on the complexity of one-time-only branching programs, *in* "Proceedings of FCT ," Lect. Notes in Comput. Sci., Vol. 199, pp. 90–99, Springer-Verlag, New York/Berlin.

[J] JUKNA, S. (1986), Lower bounds on the complexity of local circuits, *in* "Proceedings, MFCS'86," Lect. Notes in Comput. Sci., Vol. 233, pp. 440–448, Springer-Verlag, New York/Berlin.

[K1] KRAUSE, M. (to appear), Exponential lower bounds on the complexity of real-time and local branching programs, *J. Inform. Process. Cybern. (EIK)*.

[K2] KRAUSE, M. (in preparation), "Lower Bounds on the Complexity of Branching Programs," thesis.

[Ku] KUZNETSOV, S. E. (1981), Combinational circuits without null chains, *Izv. Vyssh. Uchebn. Zaved. Mat.* **5**, 56–63.

[KW] KRIEGEL, K., AND WAACK, S. (1987), Lower bounds on the complexity of real-time branching programs, *in* "Proceedings, FCT'87," Lect. Notes in Comput. Sci., Vol. 278, Springer-Verlag, New York/Berlin.

[MS] MEHLHORN, K., AND SCHMIDT, E. M. (1982), Las Vegas iis better than determinism in VLSI and distributed computing, *in* "14th STOC," pp. 30–337.

[M] MEINEL, C. (1987), The power of nondeterminism in polynomial size bounded width branching programs, *in* "Proceedings FCT'87," Lect. Notes in Comput. Sci., Vol. 278, pp. 302–309, Springer-Verlag, New York/Berlin.

[N] NECHIPORUK, E. I. (1966), A Boolean function, *Dokl. Akad. Nauk* **199**, 765–766.

[P] PAUL, W. (1977), A $2.5n$-lower bound on the combinational complexity of Boolean functions, *SIAM J. Comput.* **6**, No. 3, 427–443.

[PZ] PUDLAK, P., AND ZAK, S. (1983), Space complexity of computation, preprint, University of Prague.

[R] RAZBOROV, A. A. (1985), A lower bound on the monotone complexity of the logical permanent, *Mat. Zametki* **37**, No. 6, 887–900.

[W] WEGENER, I. (1984), On the complexity of branching programs and decision trees for clique functions, Interner Bericht der Univ. Frankfurt, 1984; *Assoc. Comput. Math.* **35** (1988), 461–471.

[Y] YAO, A. (1982), The entropic limitation of VLSI-computations, *in* "13th STOC," pp. 308–311.