INFORMATIQUE THÉORIQUE ET APPLICATIONS

JURAJ HROMKOVIČ

Communication complexity and lower bounds on multilective computations

Informatique théorique et applications, tome 33, n° 2 (1999), p. 193-212

<http://www.numdam.org/item?id=ITA_1999__33_2_193_0>

Theoretical Informatics and Applications
Theoret. Informatics Appl. **33** (1999) 193–212

COMMUNICATION COMPLEXITY AND LOWER BOUNDS ON MULTILECTIVE COMPUTATIONS *, **

JURAJ HROMKOVIČ[1]

Abstract. Communication complexity of two-party (multiparty) protocols has established itself as a successful method for proving lower bounds on the complexity of concrete problems for numerous computing models. While the relations between communication complexity and oblivious, semilective computations are usually transparent and the main difficulty is reduced to proving nontrivial lower bounds on the communication complexity of given computing problems, the situation essentially changes, if one considers non-oblivious or multilective computations. The known lower bound proofs for such computations are far from being transparent and the crucial ideas of these proofs are often hidden behind some nontrivial combinatorial analysis. The aim of this paper is to create a general framework for the use of two-party communication protocols for lower bound proofs on multilective computations. The result of this creation is not only a transparent presentation of some known lower bounds on the complexity of multilective computations on distinct computing models, but also the derivation of new nontrivial lower bounds on multilective VLSI circuits and multilective planar Boolean circuits. In the case of VLSI circuits we obtain a generalization of Thompson's lower bounds on AT^2 complexity for multilective circuits. The $\Omega(n^2)$ lower bound on the number of gates of any k-multilective planar Boolean circuit computing a specific Boolean function of n variables is established for $k < \frac{1}{2}\log_2 n$. Another advantage of this framework is that it provides lower bounds for a lot of concrete functions. This contrasts to the typical papers devoted to lower bound proofs, where one establishes a lower bound for one or a few specific functions.

AMS Subject Classification. F.2.3., F.1.1., G.2.1.

Keywords and phrases: Communication complexity, lower bounds on complexity, Boolean functions, branching programs, circuits.

* *This work has been supported by the DFG Project HR 14/3-2.*

** *An extented abstract of a part of this paper has been presented at MFCS '98.*

[1] Department of Computer Science I, Algorithms and Complexity, University of Technology, Aachen, Ahornstrasse 55, 52056 Aachen, Germany;
e-mail: hromkovic@informatik.rwth-aachen.de

1. INTRODUCTION

The communication complexity of two-party protocols has been introduced by Abelson [1] and Yao [32]. The initial goal was to develop a method for proving lower bounds on the complexity of distributed and parallel computations.

Informally [1], let $f : \{0,1\}^n \to \{0,1\}$, $n \in \mathbb{N}$, be a Boolean function over a set X of n Boolean variables, and let $\pi = (X_1, X_2)$ be a partition[2] of X. A **two-party (communication) protocol D computing f according to π** consists of two computers C_I and C_{II} with unbounded computational power. At the beginning C_I obtains an input $x : X_1 \to \{0,1\}$ and C_{II} obtains an input $y : X_2 \to \{0,1\}$. Then C_I and C_{II} communicate according to the protocol by exchanging binary messages until one of them knows the result $f(x,y)$. The cost of the protocol computation on the input (x,y) is the sum of the lengths of messages exchanged. The cost of the protocol D, $cc(D)$, is the maximum of the costs over all inputs from $z : X \to \{0,1\}$[3]. The **communication complexity of f according to π**, $cc(f,\pi)$, is the cost of the best protocol computing f according to π.

There are several ways how to define the communication complexity of a Boolean function f. The choice depends on the application considered. The most common definition used in many applications is to consider the **communication complexity of f**, $cc(f)$, as the minimum of $cc(f,\pi)$ over all "almost balanced" [4] partitions of input variables.

In the almost 20 years of its existence communication complexity has established itself as a method for proving lower bounds on several fundamental complexity measures of sequential and parallel computations. To list at least some of them we mention area complexity and area-time tradeoffs of VLSI circuits, area and depth of Boolean circuits, combinational complexity of unbounded fan-in and planar circuits, length of Boolean Formulae, size of finite automata and branching programs, and time and space complexities of Turing machines. Its success in the applications is comparable with that of Kolmogorov complexity in computability and complexity theory [14].

The simpliest standard application of communication complexity is based on the division of the hardware of the computing model considered (circuit, input tape, etc.) into two parts, in such a way, that each part contains approximately half the inputs. Obviously, such a cut corresponds to an almost balanced partition of the set of input variables. So, $cc(f)$ gives a lower bound on the amount of information that must be exchanged between these two parts of hardware. Lower bounds on the size of the hardware or on some tradeoffs between hardware size and time follow. Another standard possibility is to cut time in some discrete moment t in such way that the number of input bits read before t is approximately the same as the number of input values read after t. Again this corresponds to an almost balanced

---

[1] For a formal definition see [14] or [17].

[2] $X_1 \cup X_2 = X$ and $X_1 \cap X_2 = \emptyset$.

[3] We consider an input as an assignment of values to the input variables.

[4] Usually almost balanced means that at least one third of the input is assigned to each of the two computers of a protocol.

partition of the set of input variables. Then $cc(f)$ is a lower bound on the amount of information transferred between two time units of the computation. To realize this transfer the size of the hardware (memory, circuit) has to be large enough. The standard applications mentioned above are elegant and transparent and the main technical difficulty lies in proving nontrivial lower bounds on $cc(f)$ of a function f of interest. But these cuts with required properties can be found only if the computing model is oblivious[5] and semilective[6]. If the model is 2-multilective (each variable may enter at most twice) the ideas for the above standard applications do not work anymore, because there are no cuts corresponding to partitions of the set of input variables as defined above. This should be not surprising because usually multilective computing models are much more powerful than their semilective counterparts. To see this, consider for instance branching programs [31], VLSI circuits [22], space bounded Turing machines [14], finite automata, etc. So the known lower bound proofs for multilective computations are far from being obvious and transparent (see, for instance, [6, 8, 11, 12, 16, 19, 21, 22, 26, 28, 33]) and the crucial ideas of these proofs are often hidden behind some nontrivial combinatorial analysis.

The main aim of this paper is to create a general framework enabling to present some lower bound proofs on multilectivity as a well-structured transparent method based on two-party communication protocols. Another consequence of our effort are some new applications for proving lower bounds for multilective VLSI circuits and multilective planar Boolean circuits. Moreover, using our approach one can get lower bounds for numerous concrete functions and not only for one (or a few) function as it is usual for lower bound proofs.

The paper is organized according to three steps in which we consecutively present our method for proving lower bounds on multilective computations. Section 2 introduces the definition of a so called "overlapping" communication complexity. This captures the fact, that for multilective devices we are unable to cut them in such way, that the cut corresponds to a partition of the set of input variables X into two disjoint subsets. But what is possible to find, is a cut corresponding to a partition of X into X_1 and X_2 in such a way that $X_1 - X_2$ and $X_2 - X_1$ are "large enough". Because we have methods to prove nontrivial lower bounds on communication complexity according to such "overlapping" partitions this concept has good chances to be applied.

Section 3 is devoted to the problem how to search for a cut of a multilective device (computation). The cuts cannot be found so easily as in the semilective case. Usually we need to partition the device (computation) considered into small pieces and then to build the cut by "sticking" some small pieces together. The method explaining how to partition and how to stick together is based on a Ramsey-like combinatorial lemma. We present this lemma having broad applications in

[5]Obliviousness means that the position (time), where (when) an input enter our computing device is fixed for every input variable, *i.e.* independent of the values of specific inputs.

[6]Semilectivity means that each variable enters the computing device exactly once, *i.e.* is read exactly once.

Section 3 and explain there its relation to overlapping communication complexity introduced in Section 2.

In Section 4 we illustrate the applications of the concept developed in Sections 2 and 3 for proving lower bounds on multilective computations. In [8,12] the lower bound proofs on the area complexity of multilective VLSI circuits are presented. Our first result provides a transparent presentation of these results in terms of overlapping communication complexity. Our second result generalizes the Thompson's [25] lower bound method on the AT^2 complexity tradeoff for VLSI circuits to the multilective case. In both results mentioned above overlapping communication complexity is applied for multilective VLSI circuits in exactly the same way as communication complexity was applied for semilective VLSI circuits (see, *e.g.* [14,25,29]).

The second model considered in Section 4 is the branching program model introduced in [20]. Again we show the transparency of the application of overlapping communication complexity for proving exponential lower bounds on the size of oblivious k-times-only branching programs [15,16] for $k < \frac{1}{2}\log_2 n$ (n is the size of the input). Moreover we show that our method also works for proving exponential lower bounds on the size of oblivious branching programs with unbounded multilectivity and depth bounded by $\frac{1}{12}n\log_2 n$. Such a lower bound has been established in [2] for a specific Boolean function. Our method enables to prove lower bounds of this kind for several Boolean functions. To prove lower bounds for the general non-oblivious k-times-only branching programs seems to be an essentially harder task than proving lower bounds for the oblivious ones. In fact we can still show how overlapping communication complexity can be used to get the exponential lower bounds established in [6,19], but in this case we do not see any possibility to essentially simplify the proofs from [6,19]. Generally it seems that if one adds non-obliviousness to multilectivity the transparency of our lower bound proof method essentially decreases.

The last model we consider are multilective planar Boolean circuits. The $\Omega(n^2)$ lower bound on the number of gates of any k-multilective planar Boolean circuit computing a specific Boolean function of n variables is achieved for k being a constant independent on n. This is a generalization of the $\Omega(n^2)$ lower bounds established in [13,27] for (semilective) planar Boolean circuits. In fact we prove a more general lower bound in terms of overlapping communication complexity for $k \leq \frac{1}{2}\log_2 n$ and we extend this result to some improved lower bounds on the layout-area complexity of k-multilective planar Boolean circuits. The highest known lower bound $\Omega(n\log_2 n)$ [11,28] on the combinatorial complexity of multilective planar Boolean circuits computing a one-output function with unbounded multilectivity can be also proved due to overlapping communication complexity. One more advantage over [11,28] is that we do not need to use expanders in order to get this lower bound, and that we can prove this lower bound for a large class of Boolean functions.

2. Overlapping communication complexity

From the reasons explained in the introduction we give the formal definition of overlapping communication complexity here. After that we shortly discuss why we have defined it as follows.

Definition 2.1. Let $f:\{0,1\}^n \to \{0,1\}$ be a Boolean function defined over a set of variables $X = \{x_1, x_2, \ldots, x_n\}$, $n \in \mathbb{N}$. Let $U_0 \subseteq X$, $V_0 \subseteq X$, $|U_0| = |V_0|$ be two disjoint subsets of X. Let k be a positive integer. A pair $\pi = (\pi_L, \pi_R)$ is called a **(U_0, V_0, k)-overlapping partition of X**, if:

1. $\pi_L \cup \pi_R = X$, and
2. there exist $U \subseteq U_0 \cap \pi_L$ and $V \subseteq V_0 \cap \pi_R$ such that $U \cap \pi_R = V \cap \pi_L = \emptyset$ and $|U| \geq |U_0|/3^{2k}$, $|V| \geq |V_0|/3^{2k}$.

$\boldsymbol{Par(X, U_0, V_0, k)}$ denotes the set of all (U_0, V_0, k)-overlapping partitions of X.

The reason to consider such partitions is the following one. May be, one knows that if C_I knows values of variables from U_0 but no variable from V_0 and C_{II} knows all from V_0 but none of U_0 then the communication complexity must be large. But one is unable[7] to find a cut separating U_0 from V_0. The idea is to find a cut where at least some parts of input variables $U \subseteq U_0$ and $V \subseteq V_0$ are separated. To have a chance to prove the necessity of a long communication the sizes of U and V may not be too small in the comparison to the size of U_0 and V_0 respectively[8]. The constant 3^{2k} estimating the size of U and V depends on some combinatorial considerations given later in Lemma 3.1.

Definition 2.2. Let k be a positive integer. Let f, X, U_0, V_0 have the same meaning as in Definition 2.1. For every $\pi \in Par(X, U_0, V_0, k)$ we define the **k-overlapping communication complexity of f according to π, $occ^k(f, \pi)$**, as the complexity of the best protocol computing f according to π.

For all disjoint subsets $U_0, V_0 \subseteq X$ we define the **k-overlapping communication complexity of f according to U_0 and V_0** as

$$\boldsymbol{occ^k(f, U_0, V_0)} := \min\{occ^k(f, \pi) \mid \pi \in Par(X, U_0, V_0, k)\}.$$

Finally, the **k-overlapping communication complexity of f** is

$$\boldsymbol{occ^k(f)} := \max\{occ^k(f, U_0, V_0) \mid U_0 \subseteq X, V_0 \subseteq X, U_0 \cap V_0 = \emptyset\}.$$

In what follows we also want to apply a restricted version of overlapping communication complexity. For every $r \in \mathbb{N} - \{0\}$, we say that a communication between C_I and C_{II} has r rounds if exactly r messages have been exchanged. A protocol is called a **r-rounds**[9] protocol if for every input the communication of the protocol consists of at most r rounds. Let $\boldsymbol{occ_r^k(f, \pi)}$ be the complexity of the

[7] Usually, such cut even does not exists.

[8] Obviously, the communication complexity cannot be higher than the number of important values that could be exchanged in the communication.

[9] For formal definition and the study of r-rounds protocol see [9].

best r-rounds protocol computing f according to π for any $\pi \in Par(X, U_0, V_0, k)$. Then

$$\boldsymbol{occ_r^k(f, U_0, V_0)} := \min\{occ_r^k(f, \pi) \mid \pi \in Par(X, U_0, V_0, k)\}$$

for any disjoint subsets U_0, $V_0 \subseteq X$. The **k-overlapping r-rounds communication complexity of f** is

$$\boldsymbol{occ_r^k(f)} := \max\{occ_r^k(f, U_0, V_0) \mid U_0, V_0 \subseteq X, U_0 \cap V_0 = \emptyset\}.$$

The k-overlapping communication complexity has been introduced in order to be applied for lower bounds on k-multilective computations where each variable can be read at most k times. We see that k is strongly related to the size of the input variable subsets U and V (see Def. 2.1(2)), that are separated by a cut. With the growth of the multilectivity the sizes of subsets, one is able to separate, decrease. Why the speed-up of the decrease of $|U|$ is related to $|U_0|/3^{2k}$ we shall see in the next section. The reason to consider r-rounds protocols is that sometimes one is able to find such cuts of k-multilective devices (computations) that the information flow crossing these cuts can be described by the exchange of $r = 2k$ binary messages between the two parts given by the cuts. From [9] we know that there are Boolean functions whose r-rounds communication complexity is exponential in the $(r+1)$-rounds communication complexity. So, $occ_r^k(f)$ may be essentially larger than $occ^k(f)$ for some functions f.

Before using $occ^k(f)$ or $occ_{2k}^k(f)$ to get lower bounds on k-multilective computations we should mention, that one is able to prove high lower bounds on $occ^k(f)$. This seems to be hard because following Definition 2.2, $occ^k(f)$ is the minimum over all $\pi \in Par(X, U_0, V_0, k)$ and over the communication complexities of all protocols computing f according to π. Despite of this we have standard methods (in communication complexity theory) that can be used to prove non-trivial (even linear[10]) lower bounds on the communication complexity of concrete computing problems. In what follows we prefer to present some ideas of such methods rather than to present a detailed technical proof for one special function only.

Let f be defined over a set of input variables $X = X_1 \cup X_2 \cup X_3$, where $X_i \cap X_j = \emptyset$ for $i \neq j$ and $|X_1| \geq |X|/4$, $|X_2| \geq |X|/4$, $|X| = n$. Let the values of variables in X_3 determine which pairs $(u, v) \in X_1 \times X_2$ are in some relation (for instance have to have the same value), and so they must be somehow "compared". To prove $occ^k(f) \geq n/(4 \cdot 3^{2k})$ one may choose $U_0 = X_1$ and $V_0 = X_2$ [11]. Now we have to prove $occ^k(f, \pi) \geq n/(4 \cdot 3^{2k})$ for every $\pi \in Par(X, X_1, X_2, k)$. Let $\pi = (\pi_L, \pi_R)$ be an arbitrary (X_1, X_2, k)-overlapping partition of X. Then, there exist $U \subseteq X_1 \cap \pi_L$ and $V \subseteq X_2 \cap \pi_R$ such that $U \cap \pi_R = V \cap \pi_L = \emptyset$, and $|U|$ and $|V|$ are at least $n/(4 \cdot 3^{2k})$. Now, one can choose the set of input assignments by fixing the values of variables in X_3 in such a way that $n/(4 \cdot 3^{2k})$ different pairs from $U \times V$ have to be compared. The standard methods like the fooling set method and the rank

[10] Note that the communication complexity is at most linear.

[11] Note that $occ^k(f)$ is defined as the maximum over the choices of U_0 and V_0.

method (see for instance [4,10,14,17]) are able to establish $occ^k(f,\pi) \geq n/(4\cdot 3^{2k})$. If k is a constant independent of n, we have $occ^k(f) = \Omega(n)$.

In what follows we present a possible formalization of this approach. Let, for every $\alpha \in \{0,1\}^m$, $m \in \mathbb{N}$, and every $\delta \in \{0,1\}$, $\#_\delta(\alpha)$ denote the number of occurrences of δ in α. Let $f : \{0,1\}^{2n} \to \{0,1\}$ be a Boolean function, and let $\pi = (U,V)$ be a balanced partion of the set $U \cup V$ of the input of f. For every positive integer k, we define $F_f^k : \{0,1\}^{2n\cdot 3^{2k}\cdot 2} \to \{0,1\}$ on the set $X = U_0 \cup V_0 \cup Z \cup Q$ of Boolean variables as follows. Let $d = n \cdot 3^{2k}$, $U_0 = \{u_1,\ldots,u_d\}$, $V_0 = \{v_1,\ldots,v_d\}$, $Z = \{z_1,\ldots,z_d\}$, and $Q = \{q_1,\ldots,q_d\}$. For every input $(\alpha,\beta,\gamma,\delta) \in \{0,1\}^{4d}$ $(\alpha : U_0 \to \{0,1\}, \beta : V_0 \to \{0,1\}, \gamma : Z \to \{0,1\}, \delta : Q \to \{0,1\})$, $F(\alpha_1,\ldots,\alpha_d,\beta_1,\ldots,\beta_d,\gamma_1,\ldots,\gamma_d,\delta_1,\ldots,\delta_d) = 1$ if and only if $\#_1(\gamma_1,\ldots,\gamma_d) = \#_1(\delta_1,\ldots,\delta_d) = n$ and if $1 \leq i_1 < i_2 < \cdots < i_n \leq d$, $1 \leq j_1 < j_2 < \cdots < j_n \leq d$ are positive integers such that $\gamma_{i_1} = \gamma_{i_2} = \cdots = \gamma_{i_n} = \delta_{j_1} = \delta_{j_2} = \cdots = \delta_{j_n} = 1$, then $f(\alpha_{i_1},\alpha_{i_2},\ldots,\alpha_{i_n},\beta_{j_1},\beta_{j_2},\ldots,\beta_{j_n}) = 1$.

Lemma 2.1. *For every Boolean function $f : \{0,1\}^{2n} \to \{0,1\}$, $n \in \mathbb{N}$, and every balanced partition π of the set of input variables of f, and for every positive integer k,*

$$occ^k(F_f^k) \geq cc(f,\pi).$$

Proof of Lemma 2.1. It is sufficient to prove that, for every $\pi^k \in Par(X,U_0,V_0,k)$, $occ^k(F_f^k,\pi_k) \geq cc(f,\pi)$. Without loss of generality we assume that π divides the set of input variables of f into the first half and the second half. Let $\pi^k = (\pi_L,\pi_R)$ be a partition from $Par(X,U_0,V_0,k)$. Following Definition 2.1 there exist $U \subseteq U_0 \cap \pi_L$ and $V \subseteq V_0 \cap \pi_R$ such that $U \cap \pi_R = V \cap \pi_L = \emptyset$ and $|U| \geq |U_0|/3^{2k} = n$, $|V| \geq |U_0|/3^{2k} = n$. Let $i_1 < i_2 < \cdots < i_n \leq d$, $j_1 < j_2 < \cdots < j_n \leq d$ be positive integers such that $\{u_{i_1},u_{i_2},\ldots,u_{i_n}\} \subseteq U$ and $\{v_{i_1},v_{i_2},\ldots,v_{i_n}\} \subseteq V$. Now we fix the values of variables in Z and Q as follows:

$\gamma_{i_1} = \gamma_{i_2} = \cdots = \gamma_{i_n} = 1 = \delta_{j_1} = \delta_{j_2} = \cdots = \delta_{j_n}$ and $\gamma_s = 0 = \delta_r$ for all $s \in \{1,\ldots,d\} - \{i_1,\ldots,i_n\}$ and all $r \in \{1,\ldots,d\} - \{j_1,\ldots,j_n\}$.

Now, for every input $\alpha_1,\ldots,\alpha_d,\beta_1,\ldots,\beta_d,\gamma_1,\ldots,\gamma_d,\delta_1,\ldots,\delta_d$, $\alpha_i,\beta_i \in \{0,1\}$ for every $i = 1,\ldots,d$,

$$F_f^k(\alpha_1,\ldots,\alpha_d,\beta_1,\ldots,\beta_d,\gamma_1,\ldots,\gamma_d,\delta_1,\ldots,\delta_d)$$
$$= f(\alpha_{i_1},\alpha_{i_2},\ldots,\alpha_{i_n},\beta_{j_1},\beta_{j_2},\ldots,\beta_{j_n}).$$

So, the communication complexity of any protocol computing F_f^k according to π^k is at least the communication complexity of the best protocol for f according to π. □

In fact Lemma 2.1 claims that $occ^k(F_f^k) \geq \max\{cc(f,\pi) | \pi \text{ is a balanced partition}\}$. This is very convenient because to start to apply this approach it is sufficient to take a Boolean function that is hard for at least one partition of its set of input variables. Usually, to prove a nontrivial lower bound on $cc(f,\pi)$ for a choosen partition is a task, whose difficulty is very far from the difficulty of proving a lower bound on $cc(f)$ as the minimum over all balanced partition.

Thus, the approach described above provides linear lower bounds on overlapping communication complexity of numerous specific functions. For instance, for the balanced partition π that divides the set of input variables into the first half and the second half, the functions Equality, Inequality, Disjointness, Convolution, sum of two binary integers, etc. [14, 17] have linear communication complexity.

We call attention to the fact that one can extend the concept of overlapping communication complexity to the nondeterministic and randomized (bounded-error) ones. In the same way as in the deterministic case the lower bounds on the nondeterministic (randomized) communication complexity of a function f according to a partition π can be used as lower bounds on the overlapping nondeterministic (randomized) communication complexity of a specific function F_f. So, all following applications can be extended to the nondeterministic and randomized models without any essential change in the proofs. Because these extensions are straightforward we omit their exact formulation here.

Now, the only remaining problem is how to find cuts of multilective computations corresponding to overlapping partitions. The solution is given in the following two sections.

3. A COMBINATORIAL LEMMA

In what follows we present a Ramsey-like lemma giving a very general concept for searching for cuts of multilective computations. This lemma has been proved in several versions in the literature (see for instance [8, 14]) and so we omit the presentatioon of its proof.

Lemma 3.1. *Let m, n and k be positive integers, $m \leq n/3^{2k}$, $k < \frac{1}{2}\log_2 n$. Let U_0, V_0 be two disjoint subsets of a set X, $|U_0| \geq n$, $|V_0| \geq n$. Let $W = W_0$, W_1, $\ldots$, W_d be a sequence of subsets of X with the properties $|W_i| \leq m$ for every $i = 1, \ldots, d$ and for every $x \in X$, x belongs to at most k sets of W. Then there exist $U \subset U_0$ and $V \subset V_0$ and integers $t_0 = -1, t_1, \ldots, t_b$, $b \in \mathbb{N}$, such that the following five conditions hold:*

1. $|U| \geq n/3^{2k}$, $|V| \geq n/3^{2k}$,
2. $b \leq 2k$, $t_a \in \{1, \ldots, d\}$ *for* $a = 1, 2, \ldots, b$ *and* $t_0 < t_1 < \cdots < t_b$,
3. *if* $U \cap \left(\bigcup_{j=t_i+1}^{t_{i+1}} W_j \right) \neq \emptyset$ *for some* $i = \{0, \ldots, b-1\}$ *then*
   $V \cap \left(\bigcup_{j=t_i+1}^{t_{i+1}} W_j \right) = \emptyset$,
4. *if* $V \cap \left(\bigcup_{j=t_i+1}^{t_{i+1}} W_j \right) \neq \emptyset$ *for some* $i = \{0, \ldots, b-1\}$ *then*
   $U \cap \left(\bigcup_{j=t_i+1}^{t_{i+1}} W_j \right) = \emptyset$, *and*

5. $(U \cup V) \cap \left(\bigcup_{j=t_b+1}^{d} W_j \right) = \emptyset.$

Now we explain the relation of Lemma 3.1 to our lower bound proof concept by describing the interpretation of symbols (objects) appearing in Lemma 3.1. As before X denotes the set of input variables of a computing problem that has to be solved in a k-multilective computation of a computing device. The sets U_0 and V_0 have the same meaning as in Definition 2.1 of an overlapping partition of X. These two subsets of X one may choose arbitrarily. Obviously, the quality of the resulting lower bound essentially depends on the appropriate choice of U_0 and V_0. The fact that one may choose U_0 and V_0 is the consequence of the fact that $occ^k(f)$ is defined as the maximum over all choices of U_0 and V_0.

The idea to apply Lemma 3.1 in the search for a cut of the hardware or of the computation of a k-multilective computing device corresponding to an overlapping partition from $Par(X, U_0, V_0, k)$ is as follows. Partition the hardware (or the computation) into d "very small" pieces (or time intervals), where d may be arbitrarily large. The pieces have to be such small, that the number of variables entering one piece (the number of variables read in one interval) is bounded by $m \leq n/3^{2k}$. Obviously each variable enters (is read in) at most k different pieces (intervals). Then Lemma 3.1 says that one can stick the pieces corresponding to $W_0, W_1, \ldots, W_d$ together into at most $b+1 \leq 2k+1$ larger pieces[12]

$$\overline{W_i} = \bigcup_{j=t_i+1}^{t_{i+1}} W_j, \quad \overline{W} = \bigcup_{j=t_b+1}^{d} W_j$$

for $i = 0, 1, \ldots, b-1$ in such a way that there exist $U \subseteq U_0$ and $V \subseteq V_0$ with the properties $\overline{W_i} \cap U = \emptyset$ or $\overline{W_i} \cap V = \emptyset$ for all $i = 0, 1, \ldots, b-1$[13] and $(U \cup V) \cap \overline{W} = \emptyset$[14].

The final product is a cut of the hardware (time) into only two parts L and R, where L is the union of all parts of corresponding W_i's containing no variable from V, and R is the union of the rest (*i.e.* the complement). The cut (L, R) corresponds to a partition $\pi = (\pi_L, \pi_R) \in Par(X, U_0, V_0, k)$. Thus we have the lower bound striven for, because $occ^k(f, \pi) \geq occ^k(f, U_0, V_0)$ bits must flow via the boundary between the parts L and R.

So, Lemma 3.1 provides a general strategy for proving lower bounds on multilective computations by communication complexity. But there are a few free parameters in this strategy and these parameters decide about the success of this method. The first free parameter is the choice of U_0 and V_0. A deep analysis of the inner structure of the computing problem is necessary to find the best possibility. The second free parameter is the manner in which the hardware (time) is partitioned into small pieces corresponding to $W_0, W_1, \ldots, W_d$. The partition

[12] See (2) of Lemma 3.1.

[13] See (3) and (4) of Lemma 3.1.

[14] See (5) of Lemma 3.1.

should be done in such a way that after sticking the pieces together[15] the resulting border between L and R is as small as possible. The second free parameter does not depend on the computing problem considered, but on the multilective computing model. In the next section we illustrate the choice of this second parameter for multilective versions of VLSI circuits, planar Boolean circuits and branching programs.We do not give any example of the choice of U_0 and V_0 for a computing problem f because we rather focus on the application of the known $occ^k(f)$ for proving lower bounds on multilective computation and not on techniques proving lower bounds on $occ^k(f)$. For a careful presentation of methods for proving lower bounds on communication complexity see [4, 10, 14, 17].

4. Applications of overlapping communication complexity and Lemma 3.1 for proving lower bounds

We start to illustrate the applications on VLSI circuits. The formal definition of k-multilective VLSI circuits may be found in [14,22]. The only important points for us are the following ones:

- the circuit is laid out in a grid and the area complexity of the circuit is considered to be the size of a minimal rectangular grid in that the circuit can be laid out.
- Every input processor of the circuit may read at most one value of an input variable in one time unit.
- Every variable may enter the circuit at most k times (this means, that to every input variable x there are assigned at most k pairs from time $\times$ set of processors, where a pair (t, p) for x means that the actual value of x is read by the processor p during the t-th synchronization clock).
- Every processor computes a Boolean value in every synchronized time interval between two clocks and this value is distributed to all edges (wires) outgoing from this processor.

First we give a transparent presentation of the method introduced by Ďuriš and Galil [8] and Hromkovič *et al.* [12] for proving lower bounds on the area of k-multilective VLSI circuits. Let for a given circuit S, $A(S)$ denote the area complexity of S, $T(S)$ denote the time complexity of S, and $P(S)$ denote the number of processors of S. Obviously $P(S) \leq A(S)$ for every S.

Theorem 4.1. *Let f be a Boolean function of n' variables, for a positive integer n'. Let $k < \frac{1}{2}\log_2 n' - 2$ be a positive integer. Then, for every k-multilective VLSI circuit computing f,*

$$A(S) \geq P(S) \geq occ^k_{2k}(f)/2k.$$

Proof of Theorem 4.1. Let $\overline{X}$ be the set of input variables of f. Let U_0, $V_0 \subseteq \overline{X}$ be such that $occ^k_{2k}(f) = occ^k(f, U_0, V_0)$. So it is sufficient to prove

[15] First to $\overline{W_i}$'s and $\overline{W}$ and then to L and R.

$P(S) \geq occ^k_{2k}(f, U_0, V_0)$. Let $|U_0| = |V_0| = n$ and $X = U_0 \cup V_0$. The main point is that we cut the time of the computation into the smallest possible parts by setting

$$W_i = \{x \in X \mid x \text{ is read by } S \text{ in the } i\text{-th time unit}\}$$

for $i = 0, 1, 2, \ldots, T(S)$. Now we distinguish two possibilities according to the cardinalities of W_i's:

1. there exists $j \in \{0, 1, \ldots, T(S)\}$ such that $|W_j| > m = n/3^{2k}$. Then S must have at least $|W_j| > n/3^{2k}$ input processors, *i.e.* $P(S) > n/3^{2k}$. This completes the proof because $occ^k_{2k}(f) \leq n/(2 \cdot 3^{2k})$ for every k, n and f.
2. For every $j \in \{0, 1, \ldots, T(S)\}$, $|W_j| \leq m = n/3^{2k}$. This means, that X, $W_0, W_1, \ldots, W_{T(S)}$, k, m and n satisfy all the assumptions of Lemma 3.1. The existence of $U \subseteq U_0$, $V \subseteq V_0$, $b \in \mathbb{N}$, $t_0 = -1, t_1, \ldots, t_b \in \{1, \ldots, T(S)\}$ with the properties (1),(2),(3),(4) and (5) of Lemma 3.1 follows.
   Since an internal configuration of a circuit S (state of the circuit in a moment) can be coded as a binary word of length $P(S)$[16], the computing step from a configuration to its successor (the next configuration) can be seen as a transfer (communication) of at most $P(S)$ bits. Considering $\pi = (\pi_L, \pi_R)$ with $\pi_L = \overline{X} - U$ and $\pi_R = \overline{X} - V$[17] one can construct a $2k$-rounds protocol D according to π computing f within communication complexity $b \cdot P(S) \leq 2k \cdot P(S)$. This is because C_I knows inputs from π_L, C_{II} knows inputs from π_R, and each of them alternatively simulate the work of the circuit S in the corresponding time interval. C_I begins the simulation of S for the time interval in which no variable from U is read. Then C_{II} sends the code of the configuration of S to C_{II} and C_{II} continues in the simulation. Since the number of message exchanges in the simulation between C_I and C_{II} is at most $b \leq 2k$, and the communication complexity of the protocol D is at least $occ^k_{2k}(f)$ we obtain:

$$2k \cdot P(S) \geq b \cdot P(S) \geq occ^k_{2k}(f).$$

□

We observe that Theorem 4.1 is exactly the generalization of the known lower bound method for (semilective) VLSI circuits, where one-way communication complexity provides lower bounds on the area complexity (see, *e.g.* [14]). Now, we present a new result by showing that overlapping communication complexity may be even used to prove lower bounds on AT^2-tradeoff of k-multilective VLSI-circuits. This generalizes a similar result [25] for the relation between communication complexity and (semilective) VLSI circuits. In this case we cut the circuit (hardware) instead of the time.

[16] That contains the Boolean values computed by all processors in the last time interval.

[17] Obviously π corresponds to the cut (L, R) provided by Lemma 3.1.

Theorem 4.2. *Let k and n' be positive integers, $k < \frac{1}{2}\log_2 n' - 2$. Let f be a Boolean function depending on all its n' variables. Then, for every k-multilective VLSI program S computing f,*

$$A(S) \cdot (T(S))^2 \geq (occ^k(f)/4k)^2.$$

Proof of Theorem 4.2. Let $\overline{X}, X, U_0, V_0$ have the same meaning as in the proof of Theorem 4.1. Without loss of generality we assume the optimal layout of S has a size $u \times v$ with $u \leq v$. Let $p_0, \ldots, p_{P(S)-1}$ be the processors of f ordered lexicographically according to the layout position[18]. We choose W_i as the set of all variables read by the processor p_i in the whole computation. Again, we distinguish two possibilities according to the cardinalities of W_i's.

1. There exists a $j \in \{0, 1, \ldots, P(S) - 1\}$ such that $|W_j| > m = n/3^{2k}$. Then $T(S) \geq n/3^{2k}$ because p_j can read at most one variable in one time unit. Similarly as in the previous proof the result follows.
2. For every $j \in \{0, 1, \ldots, P(S) - 1\}$, $|W_j| \leq m = n/3^{2k}$. Then all assumptions of Lemma 3.1 are fulfilled and we get a partition of the layout of S into $b+1$ parts corresponding to $\overline{W_j}$'s and $\overline{W}$. Because the border between two such circuit parts is perpendicular to the shortest size of the layout[19], its length is at most $u + 1$. So, the final cut[20] of the layout into L and R has a border of length at most $b(u + 1) \leq 2k(u + 1)$, *i.e.* at most $2k(u + 1)$ edges crosses the cut. It means that at most $2k(u + 1)$ bits can be exchanged between L and R in one time unit. Since the number of time units is $T(S)$,

$$2k(u + 1) \cdot T(S) \geq occ^k(f)$$

and so

$$2 \cdot A(S) \cdot (T(S))^2 \geq ((u + 1) \cdot T(S))^2 \geq (occ^k(f))^2/4k^2.$$

□

Now, we consider branching programs introduced in [20]. A branching program A on a variable set $X = \{x_1, x_2, \ldots, x_n\}$ is a labelled directed acyclic graph with one source and two sinks. The sinks are labelled by Boolean constants 0 and 1. Each non-sink node is labelled by a variable $x \in X$ and has two outgoing edges labelled by 0 or 1. A computes a Boolean function $f_A : \{0,1\}^n \to \{0,1\}$ as follows. For every input $\alpha = \alpha_1\alpha_2 \ldots \alpha_n \in \{0,1\}^n$, A starts at the source node. If A is in a non-sink node labelled by x_i, $i \in \{1, 2, \ldots, n\}$, then it moves via the edge that outgoes x_i and that is labelled by α_i. The computation ends if A reaches a sink and $f_A(\alpha)$ is the label of this sink. The **size** of A is the number of non-sink nodes of A and is denoted by **size**(A). The **depth** of A is the length of the longest directed path in A. For every $j \in \{0, 1, \ldots, \text{depth}(A)\}$, the j-**th level** of A is

[18]The position (i_1, j_1) is before (i_2, j_2) if $i_1 < i_2$ or $(i_1 = i_2$ and $j_1 < j_2)$.

[19]This follows from the ordering of processors.

[20]This cut corresponds to the partition $\pi = (\overline{X} - V, \overline{X} - U)$.

the set of all nodes having the distance j from the source. The **width** of A is the maximum of the cardinalities of its levels. A branching program A is called **levelled** if every edge of A leads from the i-th level to the $(i+1)$-th level for some $i \in \{0, 1, \ldots, \text{depth}(A) - 1\}$. The main problem here is to prove exponential lower bounds on the size of branching programs computing a specific function. Since the general problem is very hard, the effort focuses on proving high lower bounds on restricted versions of branching programs.

The simpliest versions of branching programs are ordered binary decision diagrams that are in fact oblivious semilective branching programs. To prove lower bounds for them is not very hard because one-way communication complexity provides a lower bound on the size of this restricted version of branching programs. Non-oblivious semilective branching programs are called read-once-only branching programs and we have several exponential lower bounds for them (see, *e.g.* [7, 23, 24, 31, 34, 35]). In [15, 16] an exponential lower bound on the size of k-times-only oblivious branching programs has been proved. The proof does not explicitely use the method based on communication complexity. We show that by using overlapping communication complexity we will get a transparent proof of this fact, even in a more powerful form. A k-time-only oblivious branching program A is a levelled branching program, where, for every $i = 0, 1, \ldots, \text{depth}(A)$, all nodes of the i-th level read the same variable, and every variable is read at at most k different levels of A. Using overlapping communication complexity we can partially remove obliviousness and allow several variables to be read in one level. And even for this generalized model we can simply establish exponential lower bounds[21].

Theorem 4.3. *Let f be a Boolean function of n variables, $n \in \mathbb{N}$. Let k, m be positive integers such that $k < \frac{1}{2}\log_2 n - 2$ and $m \leq n/(8 \cdot 3^{2k})$. The width of every branching program reading at most m different variables on every level, asking for every variable on at most k distinct levels, and computing f is at least:*

$$2^{occ^k_{2k}(f)/2k}.$$

Proof of Theorem 4.3. The sets $W_0, W_1, \ldots, W_d$, $d \leq n \cdot k - 1$, are the sets of variables read on the corresponding levels. Using Lemma 3.1 we find a cut of the branching programs consisting of at most $b \leq 2k$ levels. Each level can be "communicated" in the logarithm of its size. □

So Theorem 4.3 enables to prove $2^{\Omega(n)}$ lower bounds on the size of k-time-only branching programs (with the above restriction) computing specific functions. Since every level can be split into a constant number of levels with a reduced number of variables read (by inserting some dummy nodes) the assumption on the number of variables read in one level can be removed from Theorem 4.3.

[21] Note that the result of Theorem 4.3 is considered to be simple from the point of view of current development in this area. But it nicely presents the transparency of our approach and we shall use it to prove a deeper result later.

The most powerful oblivious model of branching programs considered are so called (multilective) oblivious branching programs, where there is no restriction on the number of occurences of any variable in the paths of the branching program. The best lower bounds for this general model have been established in [2,5]. Alon and Maas [2] proved an exponential lower bound on the size (width) of oblivious branching programs of depth bounded by $\frac{1}{2}n \cdot \log_2 n$ for computing a specific Boolean function. For another Boolean function Babai *et al.* [5] proved an exponential lower bound on the size of oblivious branching programs with the depth restricted by $o(n(\log_2 n)^2)$. In what follows we show that the concept of overlapping communication complexity can provide lower bounds of the first type [2] for various Boolean functions. We are not able to use our concept for proving the strong lower bounds of [5]. This should be not surprising because Babai *et al.* [5] used multiparty protocols to achieve this lower bound and multiparty protocols may be considered as a generalization of overlapping communication complexity rather than as a (usually considered) generalization of standard communication complexity.

Let, for every $\alpha, \beta \in \{0,1\}$, $\text{code}(\alpha\beta) = 0$ if $\alpha = \beta = 0$, $\text{code}(\alpha\beta) = 1$ if $\alpha = \beta = 1$ and $\text{code}(\alpha\beta) = \varepsilon$ if $\alpha \neq \beta$.

Let, for every Boolean function $f : \{0,1\}^n \to \{0,1\}$, $n \in \mathbb{N}$, $f^* : \{0,1\}^{4n} \to \{0,1\}$ be defined as follows. For every $\alpha = \alpha_{1,1}\alpha_{1,2}\alpha_{2,1}\alpha_{2,2}\alpha_{3,1}\alpha_{3,2}\ldots\alpha_{2n,1}\alpha_{2n,2} \in \{0,1\}^{4n}$, $f^*(\alpha) = 0$ if the length of $\text{code}(\alpha) = \text{code}(\alpha_{1,1}, \alpha_{1,2}) \cdot \cdots \cdot \text{code}(\alpha_{2n,1}, \alpha_{2n,2})$ differs from n and $f^*(\alpha) = f(\text{code}(\alpha))$ if $|\text{code}(\alpha)| = n$.

Without loss of generality we consider that each oblivious branching program is levelled and the nodes of any level are labelled by the same input variable.

Theorem 4.4. *Let f be a Boolean function of n variables, $n \in \mathbb{N}$. Let $k < \frac{1}{2}\log_2 n - 2$ be a positive integer. Then, for every oblivious branching program A that computes f^* within a depth at most $k \cdot n$*

$$\text{width}(A) \geq 2^{occ^k_{2k}(f)/2k}.$$

Proof of Theorem 4.4. Let A be an oblivious branching program of $k \cdot n$ levels computing f^*. Let $X = \{x_{i,1}, x_{i,2} | i = 1, \ldots, 2n\}$ be the set of the input variables of f^*. Let, for $i = 1, \ldots, 2n$, l_i be the number of levels of A labelled by $x_{i,1}$ or by $x_{i,2}$. Since at most one half of l_i's may be larger than twice the average, there exist $j_1, \ldots, j_n \in \{1, \ldots, 2n\}$ such that $l_{j_d} \leq 2 \cdot \frac{k \cdot n}{2n} = k$ for $d = 1, \ldots, n$. For every $c \notin \{j_1, \ldots, j_n\}$, we set $x_{c,1} = 1$ and $x_{c,2} = 0$. Using this choice we get a k-multilective branching program computing the function f. Applying Theorem 4.3 the lower bound follows. □

Corollary 4.5. *Every branching program (or formula) computing a Boolean function f^* satisfying the assumptions of Theorem 4.4 has size at least $\Omega(n \cdot \log n)$.*

Note, that there are already known exponential lower bounds on syntactic k-times-only branching programs [BRS93, Ok93, Sue97] that are a more powerful model of branching programs than those considered in Theorem 4.3. As already mentioned above we have several exponential lower bounds for $k = 1$ (for

an overview see [7]). A syntactic k-times-only branching program is a branching program with the property that no input variable $x \in X$ appears more than k times on any directed path.

The next application we consider is to prove lower bounds on multilective planar Boolean circuits. The first superlinear lower bounds on the combinational complexity of semilective planar Boolean circuits have been independently established by Turán [27] and Hromkovič [13]. Using the Planar Separator theorem of Lipton and Tarjan [18] and our concept of overlapping communication complexity we can obtain transparent proofs of lower bounds on the complexity of k-multilective Boolean circuits for several other specific Boolean functions and for $k < \frac{1}{2}\log_2 n$. This is a direct generalization of the results in [13, 27].

Let, for every Boolean circuit S, the **combinational complexity of S, $CC(S)$**, be the number of nodes of (*i.e.* the number of gates + the number of input vertices) of S. In what follows we consider that both the indegree and the outdegree of any circuit are bounded by 2. Note, that this is no restriction because the upper bound 2 on the indegree of gates is the standard assumption for the Boolean circuit model [31], and every planar circuit S with unbounded outdegree and bounded indegree can be transformed into an equivalent planar circuit S' with bounded both indegree and outdegree and $CC(S') \leq 3 \cdot CC(S)$[22]. The standard planar Boolean circuit model is semilective, which means that the input nodes are in one-to-one correspondence with the input variables. For any positive integer k, a k-multilective planar Boolean circuit [22] has for each input variable at most k distinct input nodes.

To use overlapping communication complexity for proving lower bounds on combinational complexity of multilective planar Boolean circuits we need the following version of the Planar Separator Theorem [18]. Let $c = 2 \cdot \sqrt{2}/(1 - \sqrt{2/3})$ in what follows.

Lemma 4.6. *Let $G = (V, E)$ be a planar graph of m nodes. Let $V' \subseteq V$ be a set of special nodes. Then, by removing at most $c \cdot \sqrt{m}$ nodes of G, one can partition G into $G_1 = (V_1, E_1)$, $G_2 = (V_2, E_2)$ such that*

$$-1 \leq |V_1 \cap V'| - |V_2 \cap V'| \leq 1.$$

Proof of Lemma 4.6. The proof is a direct consequence of the Planar Separator Theorem [18] and Corollary 3.4.2.10 in [14]. □

Now, we are ready to formulate our lower bound. Let, for every positive integer k, $d_k = 4c \cdot \sqrt{2k} \cdot 3^{k+1} \cdot \frac{1}{\sqrt{2}-1}$. Note that the following result is a generalization of proving quadratic lower bounds on (semilective) planar circuits by standard communication complexity [13, 14, 27].

Theorem 4.7. *Let f be a positive integer such that $k < \frac{1}{2}\log_2 n - 2$. Then, for every k-multilective planar Boolean circuit B computing f,*

$$CC(B) \geq (occ^k(f)/d_k)^2.$$

[22] See, for instance, Observation 3.2.3.2 in [14].

Proof of Lemma 4.7. Let B be a k-multilective planar circuit computing f. The structure of B is a planar graph $G_B = (V, E)$ of a degree bounded by 4. Let $|V| = m$. Let $V' \subseteq V$ be the set of input notes of B (the nodes with the indegree 0). Using Lemma 4.6 we can partition G_B into two subgraphs with approximately $|V'|/2$ input vertices of B in each by removing at most $c \cdot \sqrt{|V|}$ nodes. Because both graphs are again planar we can continue in partitioning in order to get four subgraphs, each having approximately $|V'|/4$ of input vertices. The number of removed nodes in this second partitioning step is at most

$$c \cdot \sqrt{m_1} + c \cdot \sqrt{m_2} = c \cdot (\sqrt{m_1} + \sqrt{m_2})$$

$$\leq \max\{c \cdot \sqrt{m_1} + c \cdot \sqrt{m_2} \mid m_1 + m_2 \leq m\} \leq c \cdot \frac{2}{\sqrt{2}} \cdot \sqrt{m} = c \cdot \sqrt{2} \cdot \sqrt{m}.$$

Generally, in the h-th partitioning step we divide h graphs into $2h$ subgraphs with at most $|V'| + 1$ input vertices each. The number of removed nodes in this step is at most $c \cdot (\sqrt{2})^{h-1} \cdot \sqrt{m}$. We stop this procedure if

$$\frac{|V'|}{2^h} + 1 \leq \frac{n}{8 \cdot 3^{2k}}.$$

Since $|V'| \leq k \cdot n$, we stop after at most $r = \log_2(k \cdot 3^{2k+2})$ steps. The number of nodes removed in these r steps is at most

$$\sum_{i=1}^{r} c \cdot (\sqrt{2})^{h-1} \cdot \sqrt{m} \leq c \cdot \sqrt{m} \cdot \frac{(\sqrt{2})^{r+1} - 1}{\sqrt{2} - 1}$$

$$\leq c \cdot \frac{\sqrt{2}}{\sqrt{2} - 1} \cdot \sqrt{k} \cdot 3^{k+1}\sqrt{m} = \frac{d_k}{4} \cdot \sqrt{m}.$$

So, one can reach this partition into 2^r subgraphs by removing at most $d_k \cdot \sqrt{m}$ edges of B. Now, one can apply our combinatorial lemma. We have $W_1, W_2, \ldots W_{2^r}$, where W_i contains the input variables assigned to the input vertices in the i-th subgraph of our partition of G (B). Since $|W_i| \leq n/8 \cdot 3^{2k}$, the assumptions of the combinatorial lemma are fulfilled. Thus, putting some W_i's together one obtains a partition of B into two parts corresponding to an overlapping partition of the variables of f. Moreover, the number of edges between these two parts of B is at most $d_k \cdot \sqrt{m}$. Since every edge of B transforms exactly one Boolean value during the whole computation of B on an input, the theorem follows. □

Observe that in the previous theorem we cannot use $occ_r^k(f)$ instead of $occ^k(f)$ for any r. The reason is that any protocol simulating the communication between two Boolean circuit parts may need twice as many rounds as the depth of the circuit[23].

[23] For the technical details of the construction of a protocol simulating the information transfer between two parts of a Boolean circuit see Section 3.3.3 of [14].

Next we show that if one considers the area layout complexity of Boolean circuits instead of the combinatorial complexity of Boolean circuits, then the overlapping communication complexity may even provide higher lower bounds than those derivable from Theorem 4.6. In what follows we consider the layout of Boolean circuits into a chip (lattice rectangle) as described in [14][24]. For any Boolean circuit B, $\boldsymbol{A(B)}$ denotes the **area complexity of a Boolean circuit B.** In the general layout considered above we do not give any restriction on the layout of the input nodes. If one additionally requires that all input nodes of the given Boolean circuit lay on the border of the chip (rectangle), then we use the notation $\boldsymbol{bA(S)}$ instead of $A(S)$ for every Boolean circuit S. The next theorem generalizes the lower bound methods for the area of planar Boolean circuits established in [14] by using the standard two-party communication protocols (*i.e.*, for $k = 1$).

Theorem 4.8. *Let f be a Boolean function essentially depending on all of its n variables, $n \in \mathbb{N}$. Let k be a positive integer such that $k < \frac{1}{2}\log_2 n - 2$. Then, for every k-multilective Boolean circuit B computing f,*

(i) $A(B) \geq (occ^k(f)/2k)^2$,
(ii) $bA(B) \geq n \cdot occ^k(f)/8k$.

Proof of Theorem 4.8. (i) Let B be a k-multilective Boolean circuit computing f. Let there be a layout of B into a lattice rectangle C_B of a size $a \times b$, $a \geq b$. We distinguish two possibilities according to the size of b. If $b \geq n/3^{2k}$ then (i) follows because $occ^k(f)$ cannot be larger than $n/3^{2k}$. Let $b < n/3^{2k}$. One can consider C_B as a collection of a columns of b lattice squares. This means that every column can involve at most b input nodes. So, dividing C_B into columns we obtain $W_1, W_2, \ldots, W_a$ with $|W_i| \leq b < n/3^{2k}$ for every $i \in \{1, 2, \ldots, a\}$. Since all assumptions of our Combinatorial Lemma are satisfied we get a partition of C_B that corresponds to a k-overlapping partition of the input variables of f. The partition of C_B consists of at most $2k$ lines perpendicular to the side of the length b. So, the corresponding partition of B can be achieved by removing at most $b \cdot 2k$ edges of B. Thus,

$$b \cdot 2k \geq occ^k(f).$$

Finally,

$$a \cdot b \geq b^2 \geq (occ^k(f)/2k)^2.$$

(ii) Let S_B be such a layout of B, that all input nodes of B lay on the border of the lattice rectangle S_B of a size $a \times b$, $a \geq b$. Again, we may assume $b \leq n/3^{2k}$. Using the same argument as in (i) we get[25]

$$b \geq occ^k(f)/2k.$$

[24]The layout rules are the same as the rules used for the VLSI circuit layout.

[25]Observe, that in this case $|W_i| \leq 2$ for every $i \in \{2, 3, \ldots, a-1\}$.

Since f essentially depends on n variables, there are at least n input nodes in B. So, the border of S_B has the length at least n. This implies $a \geq n/4$. Putting the lower bounds on a and b together we obtain

$$a \cdot b \geq \frac{n}{4} \cdot occ^k(f)/2k = n \cdot occ^k(f)/8k.$$

□

In [28] Turán proved $\Omega(n \log_2 n)$ lower bound on multilective planar Boolean circuits computing a specific Boolean function. The previous highest known lower bound $\Omega(n \log n / \log \log n)$ on these planar Boolean circuits with unbounded multilectivity was established by Savage [22]. The Boolean function used in [28] is not a "natural" one because its definition uses expander graphs and superconcentrators. In what follows we show that overlapping communication complexity can be used to prove $\Omega(n \log n)$ lower bounds on multilective planar Boolean circuits for many Boolean functions.

Theorem 4.9. *Let $\{f_n\}_{n=1}^{\infty}$ be a sequence of Boolean functions where $f_n : \{0,1\}^n \to \{0,1\}$. Let $k = \lfloor \frac{1}{12} \log_2 n \rfloor$, and let $occ^k(f) \geq \frac{n}{4 \cdot 3^{2k}}$. Let, for every $n \in \mathbb{N} - \{0\}$, D_n be a multilective planar Boolean circuit computing f_n^*. Then, for every sufficiently large integer n*

$$CC(D_n) \geq \frac{1}{12} n \log_2 n.$$

Proof of Theorem 4.9. We prove Theorem 4.9 by contradiction. Let $CC(D_n) < \frac{1}{12} n \log_2 n$ for infinitely many n's. Let $X = \{x_{i,1}, x_{i,2} | i = 1, \ldots, 2n\}$ be the set of the input variables of f^*, and l_i be the number of input nodes of D_n labelled by $x_{i,1}$ or $x_{i,2}$ for $i = 1, \ldots, 2n$. Since $CC(D_n) < \frac{1}{12} n \log_2 n$, the number of input nodes of D_n is bounded by $\frac{1}{12} n \log_2 n$. So, there exist $j_1, \ldots, j_n \in \{1, 2, \ldots, 2n\}$ such that $l_{j_d} < \frac{1}{12} \log_2 n$ for $d = 1, \ldots, n$.

By setting $x_{c,1} = 1$ and $x_{c,2} = 0$ for every $c \notin \{j_1, \ldots, j_n\}$ one obtains a k-multilective planar Boolean circuit D'_n computing f. Applying Theorem 4.7 we obtain

$$\begin{aligned} CC(D_n) &\geq CC(D'_n) \geq (occ^k(f)/d_k)^2 \\ &\geq \left(\left(\frac{n}{4 \cdot 3^{2k}} \right) / b \cdot \sqrt{k} \cdot 3^k \right)^2 = \frac{n^2}{4^2 b^2 k \cdot 3^{6k}} \\ &\geq \frac{n^2}{s \cdot \log_2 n \cdot n^{\frac{1}{2} \log_2 3}} = \frac{n^{2 - \frac{1}{2} \cdot \log_2 3}}{s \cdot \log_2 n} \end{aligned}$$

for some constants b and s independent on n and k. Since $\frac{1}{2} \cdot \log_2 3 < 1$ there may exist at most finitely many n's such that

$$\frac{n^{2 - \frac{1}{2} \cdot \log_2 3}}{s \cdot \log_2 n} \leq \frac{1}{12} n \cdot \log_2 n.$$

This contradicts to our assumption that $CC(D_n) < \frac{1}{12} n \cdot \log_2 n$ for infinitely many n's. □

REFERENCES

[1] H. Abelson, Lower bounds on information transfer in distributed computations, in *Proc. 19th IEEE FOCS*, IEEE (1978) 151–158.

[2] N. Alon and W. Maas, Meanders, Ramsey theory and lower bounds for branching programs, in *Proc.27th IEEE FOCS*, IEEE (1986) 410–417.

[3] M. Ajtai, L. Babai, P. Hajnal, J. Komlós, P. Pudlák, V. Rödl, E. Szemerédi and G. Turán, Two lower bounds for branching programs, in *Proc. 18th ACM STOC*, ACM (1986) 30–38.

[4] A.V. Aho, J.D. Ullman and M. Yanakakis, On notions of information transfer in VLSI circuits, in *Proc. 15th ACM STOC*, ACM (1983) 133–139.

[5] L. Babai, N. Nisan and M. Szegedy, Multiparty protocols, pseudorandom generators for logspace, and time-space trade-offs. *J. Comput. System Sci.* **45** (1992) 204–232.

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

[7] B. Bollig and I. Wegener, A very simple function that requires exponential size read-once branching programs. *Inform. Process. Lett.* **66** (1998) 53–57.

[8] P. Ďuriš and Z. Galil, On the power of multiple read in chip. *Inform. and Comput.* **104** (1993) 277–287.

[9] P. Ďuriš, and Galil Z., Schnitger G., Lower bounds on communication complexity, in *Proc. 16th ACM STOC*, ACM (1984), 81-91.

[10] M. Dietzfelbinger, J. Hromkovič and G. Schnitger, A comparison of two lower bounds methods for communication complexity. *Theoret. Comput. Sci.* **168** (1996), 39-51.

[11] H.D. Grœger, A new partition lemma for planar graphs and its application to circuit complexity, in *Proc. FCT'91, Springer-Verlag, Lecture Notes in Computer Science* **529** (1991) 220-229.

[12] J. Hromkovič, M. Krause, Meinel and Ch., S. Waack, Branching programs provide lower bounds on the area of multilective deterministic and nondeterministic VLSI circuits. *Inform. and Comput.* **95** (1992) 117-128.

[13] J. Hromkovič, Nonlinear lower bounds on the number of processors of circuits with sublinear seperators. *Inform. and Comput.* **95** (1991) 117–128.

[14] J. Hromkovič, *Communication Complexity and Parallel Computing.* EATCS Series, Springer (1997).

[15] M. Krause, Lower bounds for depth-restricted branching programs. *Inform. and Comput.* **91** (1991) 1–14.

[16] M. Krause, Ch. Meinel and S. Waack, Separating complexity classes related to certain input oblivious logarithmic space-bounded Turing machines, in *Proc. Structure in Complexity Theory (1989)* 240–249.

[17] E. Kushilevitz and N. Nisan, *Communication Complexity.* Cambridge University Press (1997).

[18] R.J. Lipton and R.E. Tarjan, A separator theorem for planar graphs. *SIAM J. Appl. Math.* **36** (1979) 177–189.

[19] E.A. Okolnishkova, On lower bounds for branching programs. *Siberian Advances in Mathematics* **3** (1993) 152–166.

[20] P. Pudlák and S. Žák, Space complexity of computation. Techn. Report, Prague (1983).

[21] M. Sauerhoff, Lower bounds for randomized read-k-times branching programs, in *Proc. STACS'98, Lecture Notes in Computer Science* (1373) 105–115.

[22] J.E. Savage, The performance of multilective VLSI algorithms. *J. Comput. System Sci.* **29** (1984) 243–273.

[23] J. Simon and M. Szegedy, A new lower bound theorem for read-only-once branching programs and its application, J. Cai, Ed., *Advances in Computational Complexity Theory*, DIMACS Series in Discrete Mathematics and Theoretical Computer Science **13**, AMS (1993) 183–193.

[24] P. Savický and S. Žák, A large lower bound for 1-branching programs. ECCC Report TR D36-96 (1996).

[25] C.D. Thompson, Area-time complexity for VLSI, in *Proc. 11th ACM STOC*, ACM (1979) 81–88.

[26] Gy. Turán, On restricted Boolean circuits, in *Proc. FCT'89, Springer-Verlag, Lecture Notes in Computer Science* **380** (1989) 460–469.

[27] Gy. Turán, Lower bounds for synchronous circuits and planar circuits. *Inform. Process. Lett.* **30** (1989) 37–40.

[28] Gy. Turán, On the complexity of planar Boolean circuits. *Comput. Complexity* **5** (1995) 24–42.

[29] J.D. Ullman, *Computational Aspects of VLSI.* Comput. Science Press, Rockwille MD (1984).

[30] I. Wegener, *The Complexity of Boolean Functions.* Wiley-Teubner Series in Computer Science, John Wiley and Sons Ltd., and Teubner, B.G., Stuttgart (1987).

[31] I. Wegener, On the complexity of branching programs and decision trees for clique funcion. *J. Assoc. Comput. Mach.* **35** (1988) 461–471.

[32] A.C. Yao, Some complexity questions related to distributive computing, in *Proc. 11th ACM STOC*, ACM (1979) 209-213.

[33] A.C. Yao, The entropic limitations on VLSI computations, in *Proc. 11th ACM STOC*, ACM (1979) 209–213.

[34] S. Žák, An exponential lower bound for one-time-only branching programs, in Proc *MFCS '84*, Springer, Berlin, *Lecture Notes in Computer Science* **176** (1984) 562–566.

[35] S. Žák, An exponential lower bound for real-time branching programs. *Inform. and Control* **71** (1986) 87–94.

Communicated by I. Wegener.
Received May, 1998. Accepted March, 1999.