Electronic Colloquium on Computational Complexity, Report No. 48 (2000)

ECCC

Restricted Nondeterministic Read-Once Branching Programs and an Exponential Lower Bound for Integer Multiplication*

Beate Bollig**

FB Informatik, LS2, Univ. Dortmund,
44221 Dortmund, Germany
bollig@ls2.cs.uni-dortmund.de

Abstract. Branching programs are a well established computation model for Boolean functions, especially read-once branching programs have been studied intensively. In this paper the expressive power of nondeterministic read-once branching programs, i.e., the class of functions representable in polynomial size, is investigated. For that reason two restricted models of nondeterministic read-once branching programs are defined and a lower bound method is presented. Furthermore, the first exponential lower bound for integer multiplication on the size of a nondeterministic nonoblivious read-once branching program model is proven.

Keywords: Computational complexity, read-once branching programs, nondeterminism, integer multiplication

1 Introduction and Results

Branching programs (BPs) or Binary Decision Diagrams (BDDs) are a well established representation type or computation model for Boolean functions.

Definition 1. *A branching program* (BP) *or binary decision diagram* (BDD) *on the variable set* $X_n = \{x_1, \ldots, x_n\}$ *is a directed acyclic graph with one source and two sinks labelled by the constants 0 or 1, resp. Each non-sink node* (*or inner node*) *is labelled by a Boolean variable and has two outgoing edges one labelled by 0 and the other by 1. At each node* v *a Boolean function* $f_v : \{0,1\}^n \to \{0,1\}$ *is represented. A c-sink represents the constant function c. If* f_{v_0} *and* f_{v_1} *are the functions at the 0- or 1-successor of* v*, resp., and* v *is labelled by* x_i*,* f_v *is defined by Shannon's decomposition rule* $f_v(a) := \overline{a_i} f_{v_0}(a) \vee a_i f_{v_1}(a)$*. The computation path for the input* a *in a* BP G *is the sequence of nodes visited during the evaluation of* a *in* G*.*

The size of a branching program G *is equal to the number of its nodes and is denoted by* $|G|$*.* $\mathrm{BP}(f)$ *denotes the size of the smallest* BP *for a function* f*. The depth of a branching program is the maximum length of a path from the source to one of the sinks.*

* An extended abstract of this paper has been presented at MFCS 2000.

** Supported in part by DFG grant We 1066/9.

ISSN 1433-8092

The branching program size of Boolean functions f is known to be a measure for the space complexity of nonuniform Turing machines and known to lie between the circuit size of f and its $\{\wedge, \vee, \neg\}$-formula size (see, e.g., [23]). Hence, one is interested in exponential lower bounds for more and more general types of BPs (for the latest breakthrough for semantic linear depth BPs see [1]). In order to develop and strengthen lower bound techniques one considers restricted computation models.

Definition 2. *i) A branching program is called read k times* (BPk) *if each variable is tested on each path at most k times.*

ii) A BP *is called oblivious if the node set can be partitioned into levels such that edges lead from lower to higher levels and all inner nodes of one level are labelled by the same variable.*

Read-once branching programs (BP1s) have been investigated intensively. Borodin, Razborov, and Smolensky [8] have proved one of the first exponential lower bounds for BPks. For oblivious branching programs of restricted depth exponential lower bounds have been proved, e.g., by Alon and Maass [2]. Nondeterminism is one of the most powerful concepts in computer science. In analogy to the definition for Turing machines, different modes of acceptance can be studied for branching programs. The following definition of Ω-branching programs [17] summarizes the most interesting modes of acceptance.

Definition 3. *Let Ω be a set of binary Boolean operations. An Ω-branching program on the variable set $X_n = \{x_1, \ldots, x_n\}$ is a directed acyclic graph with decision nodes for Boolean variables and nondeterministic nodes. Each nondeterministic node is labelled by some function $\omega \in \Omega$ and has two outgoing edges labelled by 0 and 1, resp. A c-sink represents the constant c. Shannon's decomposition rule is applied at decision nodes. If f_{v_0} and f_{v_1} are the functions at the 0- or 1-successor of v, resp., and v is labelled by ω, the function $f_v = \omega(f_{v_0}, f_{v_1})$ is represented at v.*

Definitions of nondeterministic variants of restricted BPs are derived in a straightforward way by requiring that the decision nodes fulfill the usual restrictions as for deterministic BPs. In the following if nothing else is mentioned nondeterministic BPs means OR-BPs. The results of Borodin, Razborov, and Smolensky [8] for BPks hold (and have been stated by the authors) also for OR-BPks. Moreover, Thathachar [22] has proved an exponential gap between the size of OR-BPks and deterministic BP$(k+1)$s.

Besides this complexity theoretical viewpoint people have used branching programs in applications. Representations of Boolean functions which allow efficient algorithms for many operations, in particular synthesis (combine two functions by a binary operation) and equality test (do two representations represent the same function?) are necessary. In his seminal paper Bryant [9] introduced ordered binary decision diagrams (OBDDs) which are up to now the most popular representation for formal circuit verification.

Definition 4. *Let $X_n = \{x_1, \ldots, x_n\}$ be a set of Boolean variables. A variable ordering π on X_n is a permutation on $\{1, \ldots, n\}$ leading to the ordered list $x_{\pi(1)}, \ldots, x_{\pi(n)}$ of the variables.*

i) A π-OBDD *for a variable ordering π is a* BP *where the sequence of tests on a path is restricted by the variable ordering π, i.e., if an edge leads from an x_i-node to an x_j-node, the condition $\pi(i) < \pi(j)$ has to be fulfilled.*

ii) An OBDD *is a* π-OBDD *for some variable ordering π.*

Unfortunately, several important and also quite simple functions have exponential OBDD size. Therefore, more general representations with good algorithmic behavior are necessary. Gergov and Meinel [12] and Sieling and Wegener [21] have shown independently how read-once branching programs can be used for verification. In order to obtain efficient algorithms for many operations they define a more general variable ordering.

Definition 5. *A graph ordering is a branching program with a single sink labelled* end. *On each path from the source to the sink there is for each variable x_i exactly one node labelled x_i. A graph ordering G_0 is called a tree ordering if G_0 becomes a tree of polynomial size by eliminating the sink and replacing multiedges between nodes by simple edges.*

A graph-driven (tree-driven) BP1 *with respect to a graph ordering G_0 (tree ordering T_0), G_0-*BP1 *(T_0-*BP1*) for short, is a* BP1 *with the following additional property. For an arbitrary input $a \in \{0,1\}^n$, let $\mathcal{L}(a)$ be the list of labels at the nodes on the computation path for a in the* BP1 *and similarly let $\mathcal{L}_0(a)$ be the list of labels on the computation path for a in G_0 (T_0). We require that $\mathcal{L}(a)$ is a subsequence of $\mathcal{L}_0(a)$.*

It is easy to see that an arbitrary read-once branching program G is ordered with respect to a suitable chosen graph ordering. Sieling and Wegener [21] have shown that sometimes tree-driven BP1s have nicer algorithmic properties. The main problem for the application of graph-driven BP1s is to find a good graph ordering. The only graph ordering algorithm tested in experiments is due to Bern, Meinel, and Slobodová [3] and creates tree orderings.

Nondeterministic concepts also may be useful for applications. But one has to restrict nondeterminism in the right way or to choose an appropriate mode of nondeterminism. Partitioned BDDs (PBDDs) introduced by Jain, Bitner, Fussell, and Abraham [15] are obtained by imposing strong structural restrictions on nondeterministic read-once branching programs.

Definition 6. *A* k-PBDD (*partitioned* BDD *with k parts where k may depend on the number of variables*) *consists of k* OBDD*s whose variable orderings may be different. The output value for an input a is defined as 1 iff at least one of the* OBDD*s computes 1 on a. A* PBDD *is a k-*PBDD *for some k. The size of a k-*PBDD *is the sum of the sizes of the k* OBDD*s.*

Now, we present a new restricted nondeterministic read-once branching program model which allows us to bound the power of nondeterminism.

Definition 7. *A nondeterministic graph-driven* BP1 (*tree-driven* BP1), OR-G_0-BP1 (OR-T_0-BP1) *for short, is a nondeterministic* BP1 *where the Boolean variables labelling the decision nodes are ordered according to a graph ordering* (*tree ordering*).

In the rest of this section we motivate our results. A nondeterministic Turing machine can be simulated in polynomial time by a so-called *guess-and-verify machine*. It is an open question whether the analogous simulation exists in the context of space-bounded computation. Sauerhoff [19] has given a negative answer to this question for OR-OBDDs. The requirement that all nondeterministic nodes are at the top of the OBDD may blow-up the size exponentially. The question is still open for OR-BP1s and seems to be difficult to answer. The known lower bound techniques for OR-BP1s are not subtle enough to prove exponential lower bounds on the size of *guess-and-verify* BP1s for functions representable by OR-BP1s of polynomial size. In Section 2, we investigate the expressive power of the different restrictions of OR-BP1s, i.e., the classes of functions representable in polynomial size, in order to obtain more knowledge about their structural properties. We present an exponential lower bound for nondeterministic tree-driven BP1s for a function even representable by deterministic BP1s of polynomial size.

For a lot of restricted variants of branching programs exponential lower bounds are known. Sometimes the methods are subtle enough to prove hierarchy results. Neverthelesss, the proof of exponential lower bounds on the size of BDD models for *natural* functions is not always an easy task.

Definition 8. *Integer multiplication is the Boolean function* $\mathrm{MULT}_n{:}\{0,1\}^{2n} \to \{0,1\}^{2n}$ *that computes the product of two n-bit integers. That is,* $\mathrm{MULT}_n(x,y) = z_{2n-1}\dots z_0$ *where* $x = x_{n-1}\dots x_0$ *and* $y = y_{n-1}\dots y_0$ *and* $xy = z = z_{2n-1}\dots z_0$. $\mathrm{MULT}_{i,n}$ *computes the ith bit of* MULT_n.

For OBDDs Bryant [10] has presented an exponential lower bound of size $2^{n/8}$ for $\mathrm{MULT}_{n-1,n}$. But people working on OBDDs agree on the conjecture that the OBDD size is at least of order 2^n. From the proof of Bryant's lower bound for OBDDs it follows by a simple communication complexity argument that $\mathrm{MULT}_{n-1,n}$ cannot be computed in polynomial size by k-OBDDs, which consist of k layers of OBDDs respecting the same ordering, [4] or the various nondeterministic OBDDs [11]. Incorporating Ramsey theoretic arguments of Alon and Maass [2] and using the rank method of communication complexity Gergov [11] extends the lower bound to arbitrary linear-length oblivious BPs. It took quite a long time until Ponzio [18] was able to prove an exponential lower bound of size $2^{\Omega(n^{1/2})}$for $\mathrm{MULT}_{n-1,n}$ for BP1s. He doubts that $2^{\Theta(n^{1/2})}$ is the true read-once complexity of $\mathrm{MULT}_{n-1,n}$ but the counting technique used in his proof seems limited to this lower bound. Until now an exponential lower bound on the size of $\mathrm{MULT}_{n-1,n}$ for a nondeterministic nonoblivious branching program model is unknown. For the lower bound technique based on the rectangle method due to Borodin, Razborov, and Smolensky [8] we have to be able to count the number of 1-inputs which seems to be difficult for $\mathrm{MULT}_{n-1,n}$. In Section 3, we present an exponential lower bound for $\mathrm{MULT}_{n-1,n}$ on the size of

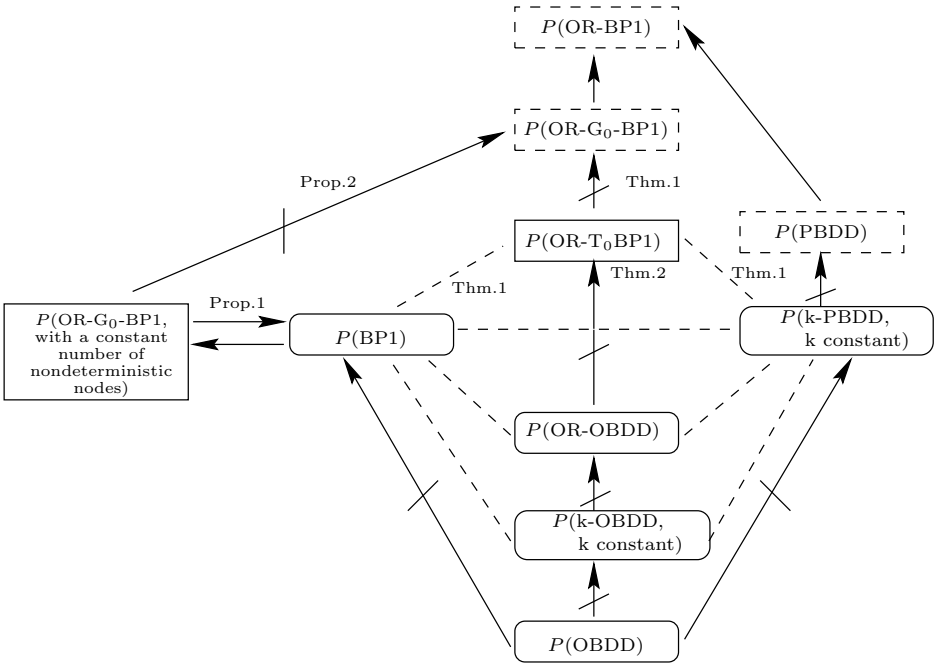

Fig. 1. *The complexity landscape for nondeterministic (ordered) read-once branching programs and the classification of* $\mathrm{MULT}_{n-1,n}$.

nondeterministic tree-driven BP1s. From this result we obtain more knowledge about the structure of MULT_n which seems to be necessary to improve the lower bounds.

Figure 1 summarizes the results (for more details see Section 2 of this paper and Section 4 of [7]). For a branching program model M we denote by $P(M)$ the class of all (sequences of) Boolean functions representable by polynomial size branching programs of type M. Solid arrows indicate inclusions and slashes through the lines proper inclusions. A dotted line between two classes means that these classes are not comparable. $P(M)$ surrounded by an oval or an rectangle means $\mathrm{MULT}_{n-1,n} \notin P(M)$. The ovals indicate known results while the rectangles indicate our new results. A dotted rectangle means that it is unknown whether the class contains $\mathrm{MULT}_{n-1,n}$. The numbers in the figure refer to the results of this paper.

2 Restricted Models of Nondeterministic Read-Once Branching Programs

Sauerhoff [19] has asked how much nondeterminism is required to exploit the full power of a computation model and how the complexity of concrete problems depends on the amount of available nondeterminism. He has investigated these

questions for OR-OBDDs and has proved that the requirement to test all nondeterministic nodes at the top, i.e., at the beginning of the computation, might blow-up the size exponentially. In order to prove an exponential lower bound for parity read-once branching programs Savický and Sieling [20] have recently presented a hierarchy result for read-once branching programs with restricted parity nondeterminism. Only at the top of the BDD parity nodes are allowed. Their result also holds (and has been stated by the authors) for OR-BP1s. For nondeterministic graph-driven read-once branching programs there cannot exist such a hierarchy result.

Proposition 1. *If a function f_n on n Boolean variables is representable in polynomial size by nondeterministic graph-driven* BP1s *with a constant number of nondeterministic nodes, f_n is contained in P*(BP1)*.*

Proof. It is easy to see that a function representable in polynomial size by a nondeterministic graph-driven BP1 with a constant number of nondeterministic nodes can also be represented in polynomial size by a nondeterministic graph-driven BP1 with a constant number of nondeterministic nodes at the top of the branching program. Let G_f be a nondeterministic graph-driven BP1 of this kind for f and k be the number of nondeterministic nodes. A binary synthesis step computing a graph-driven BP1 G_h according to a graph ordering G_0 for $h = g_1 \otimes g_2$ ($\otimes$ is a binary Boolean operation) from G_0-driven BP1s G_{g_1} and G_{g_2} for g_1 resp. g_2 can be done in time $O(|G_0| \cdot |G_{g_1}| \cdot |G_{g_2}|)$ which is also the bound for the size of G_h [21]. This result also works for k-ary ORs. Therefore, we can construct a deterministic BP1 for f whose size is bounded by $O(|G_0||G_f|^k)$. □

The function 1-VECTOR$_n$ is defined on $n \times n$ Boolean matrices X and outputs 1 iff the matrix X contains an odd number of ones and a row consisting of ones only or an even number of ones and a column consisting of ones only.

Proposition 2. *The function* 1-VECTOR$_n$ *on n^2 Boolean variables can be represented by* OR-OBDD*s of size $O(n^3)$ and by* OR-BP1*s of size $O(n^2)$ with one nondeterministic node but for* OR-G_0-BP1*s with a constant number of nondeterministic nodes the size is $2^{\Omega(n^{1/2})}$.*

Proof. Nondeterministic OBDDs are a restricted variant of nondeterministic tree-driven BP1s. It is easy to see that the function 1-VECTOR$_n$ can be represented by OR-OBDDs with $O(n)$ nondeterministic nodes in size $O(n^3)$. We can guess the row or the column consisting of ones only and check whether the number of ones in the matrix is odd or even. The size does not depend on the choice of the variable ordering.

Bollig and Wegener [7] have shown that 1-VECTOR$_n$ can be represented by 2-PBDDs of size $O(n^2)$. Obviously, PBDDs are very restricted OR-BP1s with nondeterministic nodes only at the top of the BDD. Furthermore, Bollig and Wegener have proved that deterministic BP1s need size $2^{\Omega(n^{1/2})}$. Our result follows from the proof of Proposition 1. □

Unfortunately, we are not able to prove whether there is a sequence of functions $f_n : \{0,1\}^n \to \{0,1\}$ which can be represented by OR-BP1s of polynomial size but for which OR-G_0-BP1s without restriction of the number of nondeterministic nodes require exponential size. But we conjecture that there exists such a function. The situation is different for OR-G_0-BP1 and OR-T_0-BP1.

Sieling and Wegener [21] have shown that the *hidden weighted bit function* HWB which computes x_{sum} where *sum* is the number of ones in the input needs deterministic tree-driven BP1s of exponential size but has polynomial-size deterministic graph-driven BP1s. Now, we prove that also in the nondeterministic case the expressive power of the two models is different. Using communication complexity Hromkovič and Sauerhoff [14] have presented an exponential lower bound of size $2^{\Omega(n)}$ on the size of OR-OBDDs for the function *monochromatic rows or columns* which is defined in the following way. Let X be an $n \times n$ Boolean matrix and z be a Boolean variable. Then

$$\mathrm{MRC}_n(X) := \left(z \wedge \bigwedge_{1 \le i \le n} (x_{i,1} \equiv \cdots \equiv x_{i,n}) \right) \vee \left(\overline{z} \wedge \bigwedge_{1 \le i \le n} (x_{1,i} \equiv \cdots \equiv x_{n,i}) \right).$$

Here, we investigate a very similar function $\mathrm{MRC}_n^* : \{0,1\}^{n^2} \to \{0,1\}$ which is only defined on an $n \times n$ Boolean matrix X by

$$\mathrm{MRC}_n^*(X) := \bigwedge_{1 \le i \le n} (x_{i,1} \equiv \cdots \equiv x_{i,n}) \vee \bigwedge_{1 \le i \le n} (x_{1,i} \equiv \cdots \equiv x_{n,i}).$$

We prove an exponential lower bound on the OR-OBDD size for MRC_n^* by reducing the *equality function* $\mathrm{EQ}_{n-1} : \{0,1\}^{n-1} \times \{0,1\}^{n-1} \to \{0,1\}$ to MRC_n^*. Using the fact that the nondeterministic communication complexity of EQ_{n-1} is $n-1$ it follows that $\mathrm{MRC}_n^* \notin P(\text{OR-OBDD})$. (See, e.g., [13] and [16] for the theory of communication complexity.)

Theorem 1. *There exists a function f_n on n^3 Boolean variables which needs exponential size for* OR-T_0-BP1*s but is contained in P*(BP1) *and P*(2-PBDD).

Proof. The function $f_n : \{0,1\}^{n^3} \to \{0,1\}$ is defined as disjunction of n disjoint copies of MRC_n^*. Let X_i, $1 \le i \le n$, be an $n \times n$ Boolean matrix and

$$f_n(X_1, \ldots, X_n) := \mathrm{OR}_n(\mathrm{MRC}_n^*(X_1), \ldots, \mathrm{MRC}_n^*(X_n)).$$

The proof method is the following one. We assume that f_n has nondeterministic tree-driven BP1s of polynomial size with respect to a tree ordering T_0. In T_0 there exists a path from the source to the sink which contains only $O(\log n)$ branching nodes, i.e., nodes with different 0- and 1-successor. Fixing the variables labelling these branching nodes in an appropriate way the result is a subfunction of f_n which has to be represented by a so-called nondeterministic list-driven BP1, i.e., a nondeterministic OBDD. If for all subfunctions resulting from f_n by fixing $O(\log n)$ variables by constants the size of nondeterministic OBDDs is exponential, there is a contradiction and we are done.

For any subfunction resulting from f_n by fixing $O(\log n)$ variables by constants there are $n - o(n)$ Boolean matrices X_i, $1 \leq i \leq n$, for which all variables are free, i.e., all n^2 Boolean variables of X_i are not replaced by constants. We choose one of these Boolean matrices X_i and fix all other variables not belonging to X_i in such a way that the resulting subfunction of f_n equals $\mathrm{MRC}_n^*(X_i)$. Now, we use the above mentioned lower bound.

For the first upper bound we construct a BP1 for $f_n(X_1, \ldots, X_n)$ of size $O(n^3)$. First, OR_n is represented on the pseudo variables $y_1, \ldots, y_n$ by an OBDD of size n. Afterwards, each y_i-node is replaced by a BP1 for MRC_n^* on the X_i-variables. In order to describe the BP1 for MRC_n^* we use an auxiliary tree odering T_0 which is defined in the following way. We start to test the variables according to a rowwise variable ordering. If the first row contains only 0-entries or only 1-entries, we can proceed with a rowwise variable ordering, otherwise we continue with a columnwise ordering. The width of T_0 is bounded above by $2n$. It is not difficult to see that the size of the tree ordering T_0 as well as the size of the T_0-BP1 for MRC_n^* is $O(n^2)$.

Now, we prove an upper bound of $O(n^3)$ for the 2-PBDD size of f_n. The first part checks whether there exists a matrix with monochromatic rows. All X_i-variables, $1 \leq i \leq n$, are tested one after another in a rowwise variable ordering. The second part uses a columnwise variable ordering and tests whether there is a matrix consisting of monochromatic columns. □

Proposition 2 and the proof of Theorem 1 also show that the class of functions representable by deterministic tree-driven BP1 and P(OR-OBDD) are incomparable. The function 1-VECTOR_n can be represented by OR-OBDDs of size $O(n^3)$ but even graph-driven BP1s need exponential size. On the other hand, MRC_n^* has exponential OR-OBDD size but small representations by tree-driven BP1s. Furthermore, it is not difficult to see that $P(k\text{-OBDD})$, k constant, is a proper subclass of P(OR-OBDD) which has been already stated implicitly in [7]. Proving that there are efficient algorithms for the manipulation of k-OBDDs Bollig, Sauerhoff, Sieling, and Wegener [5] have decomposed a k-OBDD representing a function f_n into an OR-OBDD of size $O(|G|^{2k-1})$ for f_n. Therefore, all functions representable by polynomial size k-OBDDs, k constant, can also be represented by OR-T_0-BP1s of polynomial size. Moreover, Bollig and Wegener [7] have shown that there are functions in P(2-OBDD) which cannot be represented by k-PBDDs, k constant, of polynomial size. It follows that there are functions in P(OR-OBDD) which cannot be represented by k-PBDDs of polynomial size.

If we relax the restriction for OR-T_0-BP1s that the deterministic variables have to be tested according to a tree ordering to the requirement that the labels of nondeterministic and deterministic nodes respect a tree ordering, we obtain a BDD model M which can represent all functions of P(PBDD) in polynomial size. But until now no function with polynomial size for OR-BP1s but exponential size for PBDDs with a polynomial number of parts is known.

3 An Exponential Lower Bound for Multiplication

We prove that nondeterministic tree-driven read-once branching programs computing integer multiplication require size $2^{\Omega(n/\log n)}$. This is the first nontrivial lower bound for multiplication on nondeterministic branching programs that are not oblivious. (See, e.g., [13] and [16] for the theory of communication complexity.)

Lemma 1. *The nondeterministic communication complexity of the problem to decide for l-bit numbers x (given to Alice) and y (given to Bob) whether $|x|+|y| \geq 2^l - c$, where c is a constant of length l with $o(l)$ ones, is $\Omega(l)$.*

Proof. We have to decide whether $|x| \geq 2^l - c - |y|$. If $c_i = 1$, we set $x_i = 0$ and $y_i = 0$. Hence, we are left with the problem to decide whether $|x^*| \geq 2^{l^*} - |y^*|$ for l^*-bit numbers x^* and y^* where $l^* = l - o(l)$. Excluding the cases $|x^*| = 2^{l^*} - 1$ and $|y^*| = 0$ this problem is identical to the decision whether $|x^*| + 1 > |z^*| := 2^{l^*} - |y^*|$. It is well-known that the nondeterministic communication complexity of the function GT_{l^*} which computes 1 for two l^*-bit numbers x' and y' iff $|x'| > |y'|$ is $\Omega(l^*)$. Using two more bits for the information whether $|x^*| = 2^{l^*}$ and $|y| = 0$ we obtain the desired lower bound of size $\Omega(l^*) = \Omega(l)$. □

Theorem 2. *The size of nondeterministic tree-driven read-once branching programs representing $MULT_{n-1,n}$ is $2^{\Omega(n/\log n)}$.*

Proof.

Using the proof method of Theorem 1 we show that for each replacement of $O(\log n)$ variables by arbitrary constants we find a subfunction of $\mathrm{MULT}_{n-1,n}$ which essentially equals the computation of the problem from Lemma 1. For this we use the ideas of Bryant's proof [10] but for our case we need some more arguments to limit the influence of the already fixed variables.

We consider an arbitrary subset of $O(\log n)$ variables and an assignment of these variables to constants. Let C_x and C_y be the sets of indices of these x- and y-variables. Variables x_j, $j \notin C_x$, and y_j, $j \notin C_y$, are called free. Let c be the result of MULT_n if we additionally set all free variables to 0 and let C be the set of indices of the 1-bits of c. Obviously $|C| = O(\log^2 n)$.

The proof idea is the following one. We decompose $x = (x_{n-1}, \ldots, x_0)$ in two numbers $x' = (x'_{n-1}, \ldots, x'_0)$ and $x'' = (x''_{n-1}, \ldots, x''_0)$ with $x = x' + x''$. The first number x' is created by setting all free x-variables to 0. For the second number x'' all variables x_i, $i \in C_x$, are set to 0. Similarly y is decomposed into y' and y''. Now, the product $z := xy$ can be written as $(x' + x'')(y' + y'')$. By definition $x'y'$ is equal to c. Our aim is to find two subvectors in x'' and y'' consisting of free bits and to replace parts of these subvectors by 0 such that the influence of the sum $x'y'' + x''y'$ to the output bit $\mathrm{MULT}_{n-1,n}$ is limited. Afterwards, we can use Bryant's proof for the rest of x'' and y''.

Now, we make these ideas precise. First, we are looking for a sequence of indices $j, j+1, \ldots, j+l-1$ of maximal length such that the input variables $x_j, \ldots, x_{j+l-1}$ and $y_{n-1-j-l+1}, \ldots, y_{n-1-j}$ are free (see Figure 3 for the choice

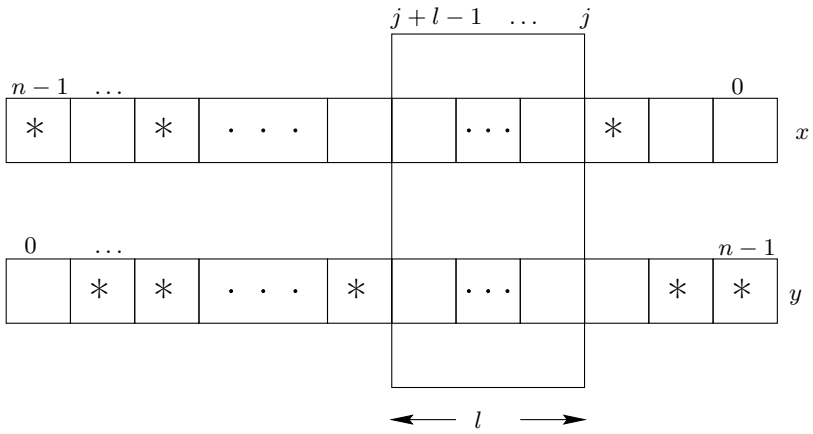

Fig. 2. *A sequence of indices $j, j+1, \ldots, j+l-1$ of maximal length such that $x_j, \ldots, x_{j+l-1}$ and $y_{n-1-j-l+1}, \ldots, y_{n-1-j}$ are free. Variables x_i, $i \in C_x$, and y_k, $k \in C_y$, are labelled by $*$.*

of the sequence). Using the pigeonhole principle we prove a lower bound on the length of such a sequence by $l = \Omega(n/\log n)$. For the ease of description we assume that l can be divided by 10. Let $X = \{x_j, \ldots, x_{j+l-1}\}$ and $Y = \{y_{n-1-j-l+1}, \ldots, y_{n-1-j}\}$ be the sets of free variables belonging to such a sequence of maximal length. We choose $X' = \{x_{j+(2/5)l}, \ldots, x_{j+(3/5)l-1}\}$. Later we set almost all variables of Y and $X \backslash X'$ to 0 to avoid an undesirable influence of the variables which are not free. In Figure 3 and 3 some of these replacements are illustrated.

Let π be an arbitrary variable ordering. The top part T of π contains the first $(1/10)l$ X'-variables with respect to π and the bottom part B the other $(1/10)l$ variables. The set of pairs $P = \{(x_{i_1}, x_{i_2}) | x_{i_1} \in T, x_{i_2} \in B\}$ has size $(1/10\, l)^2$. By a counting argument we find some set $I \subseteq \{j+(2/5)l, \ldots, j+(3/5)l-1\}$ and some distance parameter d such that $P' = \{(x_i, x_{i+d}) | i \in I\} \subseteq P$, $|P'| = |I| \geq (1/20)l$, and $\max(I) < \min(I) + d$.

We replace the variables in the following way:

- y_k is replaced by 1 for $k = n-1-\max(I)$ and $k = n-1-\max(I)-d$,
- all other free y-variables are replaced by 0,
- x_k is replaced by 1 iff $k \notin I$, $\min(i) \leq k \leq \max(I)$, and $k+(n-1-\max(I)) \notin C$,
- $x_{\max(I)+d}$ is replaced by 0 and $x_{\max(I)}$ is replaced by 1 if $n-1 \in C$, otherwise $x_{\max(I)+d}$ and $x_{\max(I)}$ are both replaced by 0,
- all other free x-variables except x_i, x_{i+d}, $i \in I$, are replaced by 0.

All the replacements are possible since all considered variables are free.

What is the effect of these replacements? Since only two free y-variables are replaced by 1, y contains these two ones and ones at the positions k where $y_k \in C_y$ and y_k is set to 1. Hence, multiplication is reduced to the addition

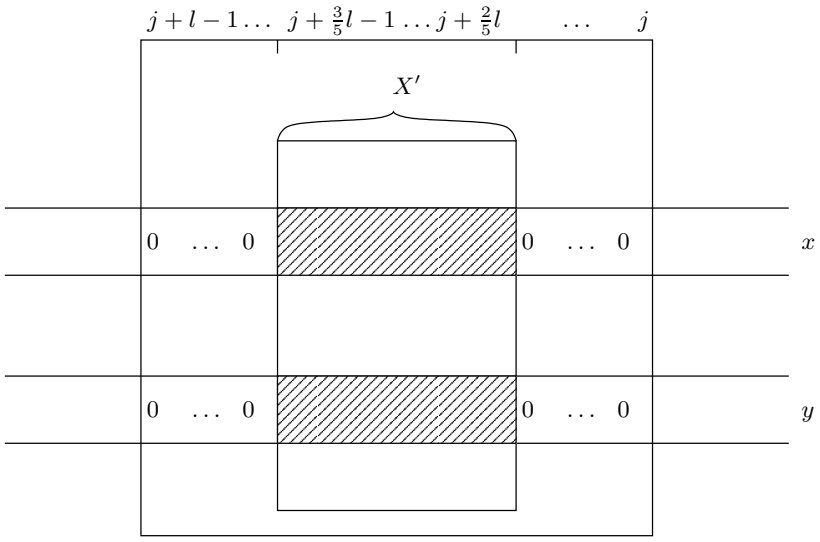

Fig. 3. *The choice of X' and some assignments to the other x- and y-variables.*

of x shifted by $n-1-\max(I)$ positions and x shifted by $n-1-\max(I)-d$ positions and x shifted by k positions if $y_k \in C_y$ and y_k is set to 1. The variables $x_j, \ldots, x_{j+(2/5)l-1}$ as well as the variables $x_{j+(3/5)l}, \ldots, x_{j+l-1}$ and all variables from Y except $y_{n-1-\max(I)}$ and $y_{n-1-\max(I)-d}$ are set to 0. Therefore, the output bit of $\mathrm{MULT}_{n-1,n}$ only depends on c and the assignments to $x_{j+(2/5)l}, \ldots, x_{j+(3/5)l-1}$, $y_{n-1-\max(I)}$, and $y_{n-1-\max(I)-d}$. Carry bits resulting from $x'y'' + x''y'$ are eliminated, since there only exist $O(\log^2 n)$ ones in C which could propagate a possible carry bit.

We are left with the situation to add two numbers and the constant c. $\mathrm{MULT}_{n-1,n}$ equals the most significant bit of this sum. Variables x_i, $i \in \{j + (2/5)l, \ldots, j+(3/5)l-1\}$ and $i \notin I$ or $i-d \notin I$, have no influence on the output bit of $\mathrm{MULT}_{n-1,n}$ since they are already replaced by constants. Together with some bits of c these variables propagate a carry if existent.

Now the result follows from Lemma 1. □

Since Lemma 1 can be extended to AND- and PARITY-nondeterminism, similar lower bounds for $\mathrm{MULT}_{n-1,n}$ can be proven for AND-T_0-BP1 and PARITY-T_0-BP1 in a straightforward way. This is the first nontrivial lower bound even for an important function on nonoblivious branching programs with an unlimited number of parity nodes. Furthermore, an extension of the proof of Theorem 2 shows that all subfunctions of $\mathrm{MULT}_{n-1,n}$ obtained by the replacement of up to $(1-\epsilon)n^{1/2}$ variables by constants have exponential size for nondeterministic tree-driven BP1.

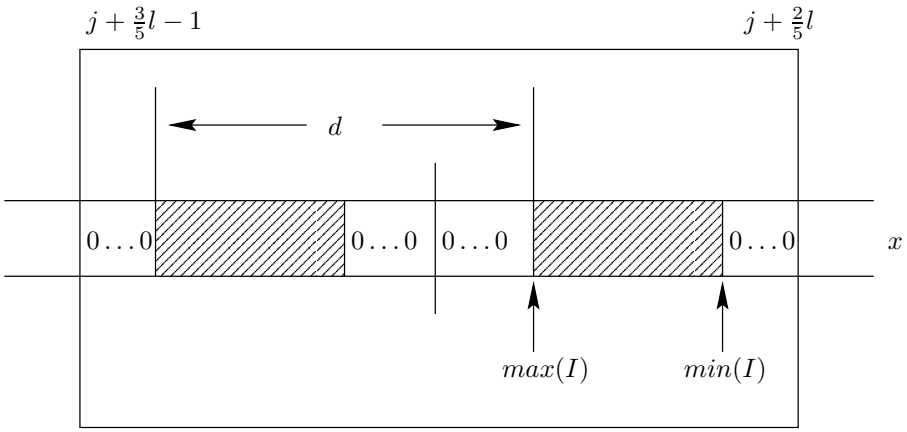

Fig. 4. *The x-vector after the replacement of some variables.*

We only want to mention that we obtain similar exponential lower bounds for the arithmetic functions squaring, inversion, and division by so-called read-once projections [6].

Acknowledgement

Thanks to Ingo Wegener for several valuable hints and fruitful discussions and to Stefan Droste for proofreading.

References

1. Ajtai, M. (1999). A non-linear time lower bound for Boolean branching programs. Proc. of 40th FOCS, 60–70.
2. Alon, N. and Maass, W. (1988). Meanders and their applications in lower bound arguments. Journal of Computer and System Sciences 37, 118–129.
3. Bern, J., Meinel, C., and Slobodová, A. (1995). Some heuristics for generating tree-like FBDD types. IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems 15, 127–130.
4. Bollig, B., Sauerhoff, M., Sieling, D., and Wegener, I. (1993). Read-k times ordered binary decision diagrams. Efficient algorithms in the presence of null chains. Tech. Report 474, Univ. Dortmund.
5. Bollig, B., Sauerhoff, M., Sieling, D., and Wegener, I. (1998). Hierarchy theorems for k-OBDDs and k-IBDDs. Theoretical Computer Science 205, 45–60.
6. Bollig, B. and Wegener, I. (1996). Read-once projections and formal circuit verification with binary decision diagrams. Proc. of 13th STACS, Lecture Notes in Computer Science 1046, 491–502.
7. Bollig, B. and Wegener, I. (1999). Complexity theoretical results on partitioned (nondeterministic) binary decision diagrams. Theory of Computing Systems 32, 487–503.

8. Borodin, A., Razborov, A., and Smolensky, R. (1993). On lower bounds for read-k-times branching programs. Comput. Complexity 3, 1–18.
9. Bryant, R. E. (1986). Graph-based algorithms for Boolean manipulation. IEEE Trans. on Computers 35, 677–691.
10. Bryant, R. E. (1991). On the complexity of VLSI implementations and graph representations of Boolean functions with application to integer multiplication. IEEE Trans. on Computers 40, 205–213.
11. Gergov, J. (1994). Time-space trade-offs for integer multiplication on various types of input oblivious sequential machines. Information Processing Letters 51, 265–269.
12. Gergov, J. and Meinel, C. (1994). Efficient Boolean manipulation with OBDDs can be extended to FBDDs. IEEE Trans. on Computers 43, 1197–1209.
13. Hromkovič, J. (1997). *Communication Complexity and Parallel Computing*. Springer.
14. Hromkovič, J. and Sauerhoff, M. (2000). Communications with restricted nondeterminism and applications to branching program complexity. Proc. of 17th STACS, Lecture Notes in Computer Science 1770, 145–156.
15. Jain, J., Abadir, M., Bitner, J., Fussell, D. S., and Abraham, J. A. (1992). Functional partitioning for verification and related problems. Brown/MIT VLSI Conference, 210–226.
16. Kushilevitz, E. and Nisan, N. (1997). *Communication Complexity*. Cambridge University Press.
17. Meinel, C. (1990). Polynomial size Ω-branching programs and their computational power. Information and Computation 85, 163–182.
18. Ponzio, S. (1998). A lower bound for integer multiplication with read-once branching programs. SIAM Journal on Computing 28, 798–815.
19. Sauerhoff, M. (1999). Computing with restricted nondeterminism: The dependence of the OBDD size on the number of nondeterministic variables. Proc. of 19th FST & TCS, Lecture Notes in Computer Science 1738, 342–355.
20. Savický, P. and Sieling, D. (2000). A hierarchy result for read-once branching programs with restricted parity nondeterminism. Preprint.
21. Sieling, D. and Wegener, I. (1995). Graph driven BDDs - a new data structure for Boolean functions. Theoretical Computer Science 141, 283–310.
22. Thathachar, J. (1998). On separating the read-k-times branching program hierarchy. Proc. of 30th Ann. ACM Symposium on Theory of Computing (STOC), 653–662.
23. Wegener, I. (1987). *The Complexity of Boolean Functions*. Wiley-Teubner.
24. Wegener, I. (2000). *Branching Programs and Binary Decision Diagrams - Theory and Applications*. SIAM Monographs on Discrete Mathematics and Applications. In print.

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