Electronic Colloquium on Computational Complexity, Report No. 46 (2000)

ECCC

New Bounds on the OBDD-Size of Integer Multiplication via Universal Hashing

Philipp Woelfel

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

Abstract. Ordered binary decision diagrams (OBDDs) nowadays belong to the most common representation types for Boolean functions. Although they allow important operations such as satisfiability test and equality test to be performed efficiently, their limitation lies in the fact, that they may require exponential size for important functions. Bryant [8] has shown that any OBDD-representation of the function $\mathrm{MUL}_{n-1,n}$, which computes the middle bit of the product of two n-bit numbers, requires at least $2^{n/8}$ nodes. This paper improves this bound to $2^{n/2}/96$ by a new technique, using a recently found universal family of hash functions [23]. As a result, one cannot hope anymore to find reasonable small OBDDs even for the multiplication of relatively short integers, since for only a 64-bit multiplication millions of nodes are required. Further, a first non-trivial upper bound of $7/3 \cdot 2^{4n/3}$ for the OBDD size of $\mathrm{MUL}_{n-1,n}$ is provided.

1 Introduction and Results

Binary Decision Diagrams (BDDs), introduced by Lee [15] and Akers [1], are a well established representation type of Boolean functions. While the general model has a large representational power and allows the simulation of any other general model of computation, its generality also has certain severe drawbacks. They lie in the NP-hardness of several important operations, that should be available for a representation serving as a dynamic data structure (see also [21]). Among these operations are e.g. the equivalence test (which tests whether two representations describe the same function), the satisfiability test (which tests whether there exists a satisfying input for the represented function), or the minimization problem (which is to minimize the size of the representation of a given function).

In order to overcome these drawbacks, Bryant [7] has introduced restricted BDDs, called *ordered binary decision diagrams* (OBDDs).

Definition 1 Let $X_n = \{x_1, \ldots, x_n\}$ be a set of Boolean variables.

1. A *variable ordering* π on X_n is a permutation of the indices $\{1, \ldots, n\}$, leading to the ordered list $x_{\pi(1)}, \ldots, x_{\pi(n)}$ of the variables.

ISSN 1433-8092

2. A π-OBDD on X_n for a variable ordering π is a directed acyclic graph with the following properties: Each sink is labeled by a constant 0 or 1. Each inner node is labeled by a variable from X_n and has two outgoing edges, one of them labeled by 0, the other by 1. If an edge leads from a node labeled by x_i to a node labeled by x_j, then $x_{\pi^{-1}(i)} < x_{\pi^{-1}(j)}$. This means that any path on the graph passes the nodes in an order respecting the variable ordering π.

3. A node v of a π-OBDD is said to compute a Boolean function $f_v : \{0,1\}^n \to \{0,1\}$, if for any $a = (a_1, \ldots, a_n) \in \{0,1\}^n$, the path starting from v and leading from any x_i node over the edge labeled by the value of a_i, finishes at the sink with label $f(a)$. A π-OBDD with a root v is said to compute a Boolean function f, if v computes f.

4. The size of a π-OBDD is the number of its nodes. The π-OBDD size of a Boolean function f (short: $\pi - \mathrm{OBDD}(f)$) is the size of the minimum π-OBDD computing f. Finally, the OBDD size of f (short: $\mathrm{OBDD}(f)$) is the minimum of $\pi - \mathrm{OBDD}(f)$ for all variable orderings π.

Efficient algorithms on π-OBDDs are known generally for all important operations, as e.g. the ones mentioned above (for an in-depth discussion of OBDDs and their operations see [21]). The size of a π-OBDD though, can be quite sensitive to the chosen variable ordering π, and finding a variable ordering leading to small or even minimal π-OBDDs is a hard problem (see [4, 7, 18]). Furthermore, since there exist 2^{2^n} Boolean functions of n variables, it can be seen that almost all functions require an exponential number of elements in any realization by networks using only primitive elements. However, this still is not disturbing as long as for all practical relevant families of functions small representations of a certain kind exist. Therefore, one is interested in finding exponential lower bounds for the size of OBDDs (and other representation types) computing important families of functions, such as integer multiplication.

Definition 2 The Boolean function $\mathrm{MUL}_{k,n} : \{0,1\}^{2n} \to \{0,1\}$ computes the bit z_k of the product $(z_{2n-1} \ldots z_0)$ of two integers $(y_{n-1} \ldots y_0)$ and $(x_{n-1} \ldots x_0)$.

The first step towards bounding the size of OBDDs for integer multiplication was done by Bryant in 1986 [7]. He showed, that for any variable ordering π, there exists an index k, such that the π-OBDD size for $\mathrm{MUL}_{k,n}$ is at least $2^{n/8}$. This result though, would still allow the possibility that one might obtain polynomial size OBDDs for all functions $\mathrm{MUL}_{k,n}$ by choosing different variable orderings for different output bits. In 1991, Bryant found that computing the middle bit of multiplication (that is $\mathrm{MUL}_{n-1,n}$) requires a π-OBDD containing $2^{n/8}$ nodes for any variable ordering π [8]. More precisely, he showed that for any $1 \leq k \leq n$, $\mathrm{OBDD}(\mathrm{MUL}_{k-1,n})$ and $\mathrm{OBDD}(\mathrm{MUL}_{2n-k-1,n})$ have a value of at least $2^{k/8}$.

Although this proves the exponential size of OBDDs for multiplication, the bound is - as stated e.g. by Bollig and Wegener in [5] - not satisfactory. This is

because Bryant's bound would still allow the possibility that one can construct 64-bit multipliers represented by OBDDs containing only 256 nodes, while on the other hand it is widely conjectured that OBDDs computing $\mathrm{MUL}_{n-1,n}$ have a size of at least 2^n. This would mean, that such a multiplier could not even be realized with millions of nodes. Since one would like to use OBDDs for "realistic" applications, such as verification of multipliers, one is interested in either finding such small constructions or a better lower bound. The following result, which is proven in the next two sections, provides the second alternative:

Theorem 1 *Any OBDD computing* $\mathrm{MUL}_{n-1,n}$ *has a size of at least* $2^{\lfloor n/2 \rfloor}/96$.

This bound shows that any OBDD for 64-bit multiplication must be constructed of more than 44 million nodes and thus demonstrates a true weakness of the representation type. The technique leading to this result is new. It highly relies on a recently found universal family of hash functions [23] and makes use of a new lemma showing how such functions distribute arbitrary sets over the range. Universal hashing is introduced in the next section.

Since it is generally believed though, that the true bound on the OBDD size for $\mathrm{MUL}_{n-1,n}$ is still larger, it is of interest to have an upper bound, too. Note, that for any Boolean function on m variables, there exists an OBDD of size $\big(2+o(1)\big)2^m/m$ [6], so a trivial upper bound for $\mathrm{OBDD}(\mathrm{MUL}_{n-1,n})$ is roughly $2^{2n}/n$. The following upper bound, proved in section 4, is the first non-trivial one.

Theorem 2 *There exists an OBDD for* $\mathrm{MUL}_{n-1,n}$ *having a size of* $7/3 \cdot 2^{4n/3}$.

The bound shows, that the middle bit of a 16-bit multiplication can be computed by an OBDD containing less than 6.2 million nodes. This is remarkable, since approaches to build OBDDs from circuits for verifying purposes have not been successful for 16-bit multipliers, yet.

2 Universal Hashing

The concept of universal hashing was introduced by Carter and Wegman in 1979 [9]. While one of its original purposes was to use randomization in hashing schemes instead of relying on the distribution of the inputs, it has found over the years a large variety of applications in areas of all different kinds. They range from algorithms for the various types of hashing based dictionaries [9, 12, 13] over cryptographic aspects as message authentication [2, 17] up to complexity theoretical statements [14, 20].

Universal hash families are usually defined by using the following notation: Let $\mathcal{H}$ be a family of hash functions $U \to R$. U and R are called *universe* and *range*, respectively. For arbitrary $x, x' \in U$ and $h \in \mathcal{H}$, we define

$$\delta_h(x,x') \;=\; \begin{cases} 1 & \text{if } x \neq x' \text{ and } h(x) = h(x') \\ 0 & \text{otherwise.} \end{cases}$$

If one or more of h, x and x' are replaced in $\delta_h(x,x')$ by sets, then the sum is taken over the elements from these sets. E.g., for $H \subseteq \mathcal{H}$, $V \subseteq U$ and $x \in U$, $\delta_H(x,V)$ means

$$\sum_{h\in H}\sum_{x'\in V}\delta_h(x,x').$$

Definition 3 A family $\mathcal{H}$ of hash functions $U \to R$ is *universal*, if for any two distinct $x, x' \in U$

$$\delta_{\mathcal{H}}(x,x') \;\leq\; \frac{|\mathcal{H}|}{|R|}.$$

A stronger definition of so-called "strongly universal hash families" was given in [22]. Among the many applications, there were also interesting results concerning branching programs (or equivalently BDDs). So, Mansour, Nisan and Tiwari [16] investigated the computational complexity of strongly universal hashing, and gave a lower bound for the time-space tradeoff of branching programs computing the functions of such families. Further, Beame, Tompa and Yan [3] have found results on the communication-space tradeoff of strongly universal hash families in a general model of two communicating branching programs.

For OBDDs, it is not possible to show a general exponential lower bound for universal hash families. E.g., the convolution of two bit strings can be viewed as a strongly universal hash family [16], but it can be easily seen that for any output bit of the convolution, there exists a variable ordering π leading to a linear size π-OBDD.

The property of universal hash families we will use here, can be described in the following way: If there are two large enough subsets of the universe given, then there exists a hash function, so that the function values of the elements from each each set cover a large fraction of the range. This is in a way a twisted version of the known results, telling that there exists a function under which the number of collisions of elements from a set is small.

For a function $h : U \to R$ and a subset $M \subseteq U$, define $h(M)$ to be the image of M under h, namely

$$h(M) := \{y \in R \mid \exists x \in M : h(x) = y\}.$$

Lemma 1 *Let $\mathcal{H}$ be a universal family of hash functions $U \to R$ and $0 \leq \epsilon < 1$. Then for arbitrary $M, N \subseteq U$ with*

$$|M| = |N| \;>\; 2\big(|R|-1\big)\frac{\epsilon}{1-\epsilon},$$

there exists a hash function $h \in \mathcal{H}$ such that $h(M)$ and $h(N)$ contain more than $\epsilon|R|$ elements each.

Proof: Let $r = |R|$, $m = |M| = |N|$ and for $h \in \mathcal{H}$ let the random variable X_h be the sum of $\delta_h(M,M)$ and $\delta_h(N,N)$. By the universal property of $\mathcal{H}$, we

obtain for a randomly chosen function h an upper bound for the expectation of X_h:

$$\begin{aligned}\mathop{\mathbf{E}}_{h\in\mathcal{H}}[X_h] &= \sum_{\substack{x,x'\in M\\ x\neq x'}} \mathop{\mathbf{E}}_{h\in\mathcal{H}}[\delta_h(x,x')] + \sum_{\substack{y,y'\in N\\ y\neq y'}} \mathop{\mathbf{E}}_{h\in\mathcal{H}}[\delta_h(y,y')] \\ &\leq |M|\big(|M|-1\big)\frac{1}{r} + |N|\big(|N|-1\big)\frac{1}{r} \\ &= \frac{2}{r}\,m(m-1).\end{aligned}$$

This means by the probabilistic method, that there exists an $h_0 \in \mathcal{H}$ with

$$X_{h_0} \leq \frac{2}{r}\,m(m-1). \tag{1}$$

In order to prove that this h_0 fulfills the claim, we assume that $h_0(M)$ contains at most ϵr elements. By summing over the ordered pairs of elements in ${h_0}^{-1}(y) \cap M$ for each $y \in h_0(M)$, we get

$$\begin{aligned}\delta_{h_0}(M,M) &= \sum_{y\in h_0(M)} \big|{h_0}^{-1}(y)\cap M\big|\Big(\big|{h_0}^{-1}(y)\cap M\big|-1\Big) \\ &= \sum_{y\in h_0(M)} \big|{h_0}^{-1}(y)\cap M\big|^2 - |M|.\end{aligned}$$

Clearly, the last sum takes its minimum, if each ${h_0}^{-1}(y)\cap M$ contains the same number of $|M|/(\epsilon r)$ elements. Therefore,

$$\delta_{h_0}(M,M) \;\geq\; \epsilon r\Big(\frac{m}{\epsilon r}\Big)^2 - m \;=\; m\Big(\frac{m}{\epsilon r}-1\Big).$$

For N, we obtain with similar arguments that $\delta_{h_0}(N,N) \geq m(m/r-1)$. So we have the following lower bound on X_{h_0}:

$$X_{h_0} \;\geq\; m\Big(\frac{m}{\epsilon r}+\frac{m}{r}-2\Big).$$

Together with the upper bound from (1), we obtain

$$\frac{2}{r}\,m(m-1) \;\geq\; m\Big(\frac{m}{\epsilon r}+\frac{m}{r}-2\Big).$$

By the assumption that $m > 2(r-1)\epsilon/(1-\epsilon)$, this results into the contradiction

$$2-\frac{2}{r} \;\geq\; m\,\frac{1-\epsilon}{\epsilon r} \;>\; 2\frac{r-1}{r}.$$

■

We consider now hash functions, which map the n-bit universe $U := \{0,\ldots,2^n-1\}$ to the k-bit range $R_k := \{0,\ldots,2^k-1\}$. For $a,b \in U$ let

$$h^k_{a,b} : U \to R_k, \quad x \mapsto \big((ax+b) \bmod 2^n\big) \operatorname{div} 2^{n-k},$$

where "div" is the integer division (i.e. $x \operatorname{div} y = \lfloor x/y \rfloor$). In a bitwise view, the result of the modulo operation $x \bmod 2^n$ is represented by the n least significant bits of x. On the other hand, the division $x \operatorname{div} 2^{n-k}$ can be seen as shifting x by $n-k$ digits to the right. In other words, if the value of the linear function $ax+b$ is represented by $(y_{2n-1} \dots y_0)$, then $h^k_{a,b}$ is the integer, that is represented by the k bits $(y_{n-1} \dots y_{n-k})$. The following result has recently been established by the author [23].

Theorem 3 *Let $1 \leq k \leq n$. There exist subsets $A, B \subseteq U$ such that the family of hash functions $h^k_{a,b}$ with $a \in A$ and $b \in B$ is universal.*

Similar hash classes have been investigated in [10, 11].

3 Lower Bounds

Since the functions $h^k_{a,b}$ are evaluated not only by a multiplication, but also by an addition, we cannot use Lemma 1 for the lower bound proof of OBDD($\mathrm{MUL}_{k,n}$) directly. Let $f^k_a := h_{a,0}$ be the functions, that can be evalutated without addition. The following lemma gives a result similar to that of Lemma 1. Note, that as stated in [11], the hash functions f^k_a form an "almost" universal hash class (which means, that in Definition 3 $|\mathcal{H}|/|R|$ is replaced by $c|\mathcal{H}|/|R|$ for some constant c). This property though, is not sufficient to prove a result as strong as the one given below.

Lemma 2 *Let $M, N \subseteq U$ and $1/2 \leq \epsilon < 1$. If $|M| = |N| \geq 2(2^{k+1}-1)\epsilon/(1-\epsilon)$, then there exists an $a \in U$, such that $f^k_a(M)$ and $f^k_a(N)$ contain at least $(2\epsilon-1)2^k$ elements each.*

Proof: By Lemma 1 and Theorem 3, there exist $a, b \in U$ such that $h^{k+1}_{a,b}(M)$ and $h^{k+1}_{a,b}(N)$ contain at least $\epsilon|R_{k+1}| = \epsilon 2^{k+1}$ elements each. Let these a, b be fixed and $f := f^k_a$. We show that $f(M)$ contains at least $(2\epsilon - 1)2^k$ elements; the claim then follows for N with the same argument.

Clearly, there exists a subset $M' \subseteq M$ with $|M'| = \epsilon 2^{k+1}$, such that all $x \in M'$ have distinct function values under $h^{k+1}_{a,b}$. Let $b = \lambda 2^{n-k-1} + \tau$, where $\lambda < 2^{k+1}$ and $\tau < 2^{n-k-1}$. We first show that replacing b by τ makes no difference in the sense that all $x \in M'$ have distinct function values under $h^{k+1}_{a,\tau}$, too. Assume that there are two different $x, x' \in M'$ with $h^{k+1}_{a,\tau}(x) = h^{k+1}_{a,\tau}(x') = z$. Then $(ax + \tau) \bmod 2^n$ and $(ax' + \tau) \bmod 2^n$ are both contained in $\left\{z2^{n-k-1}, \dots, (z+1)2^{n-k-1} - 1\right\}$. Therefore $(ax + b) \bmod 2^n$ and $(ax' + b) \bmod 2^n$ are both contained in $\left\{(z+\lambda)2^{n-k-1}, \dots, (z+\lambda+1)2^{n-k-1} - 1\right\}$, which is a contradiction, since then x and x' both have the same function value of $z + \lambda$ under $h^{k+1}_{a,b}$.

We have shown so far, that $\left|h^{k+1}_{a,\tau}(M')\right| = |M'| = \epsilon 2^{k+1}$. Next, we have to replace τ by 0. In order to do so, we assume first that $\tau < 2^{n-k-2}$. Since R_{k+1} contains exactly 2^k even elements, there are at least $|M'| - 2^k$ elements in

M', which have an odd function value under $h^{k+1}_{a,\tau}$. Let M'' be a subset of M' containing exactly $\epsilon 2^{k+1} - 2^k = 2^k(2\epsilon - 1)$ elements with an odd function value. To prove the claim, it suffices to show that for any two distinct $x, x' \in M''$ we have $f(x) \neq f(x')$. Let $h^{k+1}_{a,\tau}(x) = z$ and $h^{k+1}_{a,\tau}(x') = z'$, thus

$$z2^{n-k-1} \;\leq\; (ax + \tau) \bmod 2^n \;<\; (z+1)2^{n-k-1}.$$

Since $\tau < 2^{n-k-2}$, it follows that

$$(z - 1/2)2^{n-k-1} \;\leq\; (ax) \bmod 2^n \;<\; (z+1)2^{n-k-1}.$$

Since $z/2 = \lfloor z/2 \rfloor + 1/2$ - which is because z is odd - this implies

$$\lfloor z/2 \rfloor 2^{n-k} \leq\; (ax) \bmod 2^n \;<\; (\lfloor z/2 \rfloor + 1)2^{n-k}.$$

This means that $f(x) = \lfloor z/2 \rfloor$, and with the same argument also $f(x') = \lfloor z'/2 \rfloor$. But because z and z' are both odd and different, clearly $\lfloor z/2 \rfloor$ and $\lfloor z'/2 \rfloor$ are different, too. So, we obtain the desired result $f(x) \neq f(x')$.

Finally, we have to consider the case that $\tau \geq 2^{n-k-2}$. It can easily be verified that this case is symmetric to the former one, if M'' is chosen to be a $2^k(2\epsilon - 1)$-element subset of M' containing only elements with an even function value under $h^{k+1}_{a,\tau}$ (such an M'' exists by the same reasons as above). Then again $f(M'') = 2^k(2\epsilon - 1)$, which completes the proof. ∎

We are now ready to prove an intermediate result, from which the lower bound for the OBDD size of $\mathrm{MUL}_{n-1,n}$ follows easily. In order to do so, we have to introduce some more notation. Let x be an integer represented in a bitwise notation as $(x_{n-1} \ldots x_0)$. Then we write $[x]_k$ for the $(k+1)$-th bit x_k. Further, let $\mathrm{MUL}^a_{k,n} : \{0,1\}^n \to \{0,1\}$ for $a \in \{0,1\}^n$ be the Boolean function, that computes the $(k+1)$-th bit of the product of a with an n-bit number, i.e. $\mathrm{MUL}^a_{k,n}(x) = \mathrm{MUL}_{k,n}(a, x)$.

Theorem 4 *Let π be an arbitrary variable ordering on X_n. Then there exists an $a \in \{0, \ldots, 2^n - 1\}$ for which any π-OBDD for $\mathrm{MUL}^a_{n-1,n}$ consists of at least $2^{\lfloor n/2 \rfloor}/192 + 1$ nodes.*

Proof: Let the input variables for the π-OBDD be $x_{n-1}, \ldots, x_0$ for an n, which is w.l.o.g. even. Consider the top part T of π, which contains the first $n/2$ variables with respect to π and the bottom part B containing the other $n/2$ variables. We construct now two sets M and N of numbers in $\{0, \ldots, 2^n - 1\}$ as follows: M contains all numbers which can be represented by $(x_{n-1} \ldots x_0)$ if the variables from T are set to 0, and N contains all numbers which can be represented by $(x_{n-1} \ldots x_0)$ if the variables from B are set to 0. Note, that any number in $\{0, \ldots, 2^n - 1\}$ can be uniquely expressed as $p + q$ for $p \in M$ and $q \in N$.

Our goal is to find an appropriate constant a and two subsets $M' \subseteq M$, $N' \subseteq N$ with the following property: For any distinct q, q' in N', there exists such an $p \in M'$ that $a(p+q)$ and $a(p+q')$ differ in the n-th bit. More formally

$$\forall q, q' \in N', q \neq q' \;\exists p \in M' : \quad \big[a(p+q)\big]_{n-1} \;\neq\; \big[a(p+q')\big]_{n-1}. \qquad (2)$$

Since q and q' are determined only by the top variables and p is determined by the bottom variables, it follows that among the $2^{n/2}$ subfunctions obtained by replacing the top variables with constants, there are at least $|N'|$ pairwise different ones. So, at level $n/2$, the π-OBDD consists of at least $|N'|$ nodes. Further, a simple inductive argument shows, that any OBDD contains in a level i at most one more node than there are nodes in all preceding levels $1, \ldots, i-1$ together. Therefore, the total number of nodes in the OBDD for $\mathrm{MUL}^a_{n-1,n}$ is at least $2|N'|+1$ (including the two sinks).

Let $\epsilon = 11/12$ and $k = n/2 - 6$. Then by an easy calculation one obtains, that

$$|M| \;=\; |N| \;=\; 2^{n/2} \;>\; 2(2^{k+1}-1)\frac{\epsilon}{(1-\epsilon)}.$$

By Lemma 2, there exists an a for which $f^k_a(M)$ and $f^k_a(N)$ contain at least $(2\epsilon - 1)2^k = 5/6 \cdot 2^k$ elements each. We fix this a, define $f = f^k_a$ and continue to determine appropriate M' and N'.

As an intermediate step, we choose M^* and N^* to be minimal subsets of M respectively N, such that $f(M^*)$ and $f(N^*)$ contain exactly $2/3 \cdot 2^{k-1}$ even elements. Such sets exist, since at most 2^{k-1} of the 2^k possible function values are odd, and thus at least $5/6 \cdot 2^k - 2^{k-1} = 2/3 \cdot 2^{k-1}$ of the elements in M respectively N have distinct and even function values under f. Note, that because we required M^* and N^* to be minimal, no two elements from M^* respectively N^* have the same function value under f.

The following observation is crucial for the rest of the proof: For any $p \in M^*$ and any $q \in N^*$, the k-th bit of $f(p) + f(q)$ has the same value as the n-th bit of $a(p+q)$. Or formally

$$[f(p) + f(q)]_{k-1} = [a(p+q)]_{n-1}. \tag{3}$$

The reason for this is, that the rightmost bits of $f(p)$ and $f(q)$ are both zero (since these values are even). Recalling that the division executed by f is in fact a right-shift by $n-k$ bits, we obtain $[ap]_{n-k} = [aq]_{n-k} = 0$. Therefore, the bits of $ap + aq$ with higher index than $n-k$ are not influenced by a carry bit resulting from the addition of the less significant bits $([ap]_{n-k} \ldots [ap]_0) + ([aq]_{n-k} \ldots [aq]_0)$. This means, that $f(p) + f(q)$ has in all bits (except possibly the least significant one) the same value as $a(p+q)$ in the bits with indices $n-1, \ldots, n-k$, and equation (3) is true.

In order to satisfy property (2) it is sufficient by the above arguments that the sets M' and N' are subsets of M^* and N^* and fullfill the following:

$$\forall q, q' \in N', q \neq q' \; \exists p \in M' : \; [f(p) + f(q)]_{k-1} \neq [f(p) + f(q')]_{k-1}. \tag{4}$$

We set $M' = M^*$ and

$$N' \;=\; \left\{ q \in N^* \;\middle|\; \exists p \in M' : f(q) = 2^k - f(p) \right\}. \tag{5}$$

In order to prove claim (4), let q and q' be arbitrary distinct elements from N'. Since q and q' are in N^* and therefore have distinct function values under f,

we may assume w.l.o.g. that

$$0 \;<\; \big(f(q) - f(q')\big) \bmod 2^k \;\leq\; 2^{k-1} \tag{6}$$

(otherwise we achieve this by exchanging q and q'). By construction, there exists a $p \in M'$ with $f(p) + f(q) = 2^k$. For this p, obviously the k-th bit of $f(p) + f(q)$ equals 0. But on the other hand, by inequation (6), the value of $\big(f(p) + f(q')\big) \bmod 2^k$ is in $\{2^{k-1}, \ldots, 2^k - 1\}$. This means, that the k-th bit of $f(p) + f(q')$ equals 1, and thus claim (4) is proven.

So far, we have constructed subsets $M' \subseteq M$ and $N' \subseteq N$, which satisfy claim (2), implying by our arguments a lower bound on the π-OBDD size of $2|N'| + 1$. All that is left to do, is to give an appropriate lower bound on $|N'|$. Recall the definition of N' in (5), and that $f(M')$ and $f(N^*)$ contain $2/3 \cdot 2^{k-1}$ even elements each. Because for any even $f(p)$ also $2^k - f(p)$ is even, the set $\{2^k - f(p) \mid p \in M'\}$ contains $2/3 \cdot 2^{k-1}$ even elements, too. But since there exist only 2^{k-1} even elements in $\{0, \ldots, 2^k\}$, the intersection of $f(N^*)$ and $\{2^k - f(p) \mid p \in M'\}$ - which is $f(N')$ - has a cardinality of at least $1/3 \cdot 2^{k-1}$. By the choice of k, $2|N'|+1$ (and thus also the size of the π-OBDD) is bounded below by

$$2\big|f(N')\big| + 1 \;\geq\; 2 \cdot \frac{1}{3} \cdot 2^{k-1} + 1 \;=\; \frac{1}{3} \cdot 2^{n/2-6} + 1 \;=\; \frac{2^{n/2}}{192} + 1 \qquad \blacksquare$$

This theorem shows the general result for $\mathrm{MUL}_{n-1,n}$ by the following straightforward observation: If for some constant B and some variable ordering π there exists an a, for which $\pi - \mathrm{OBDD}(\mathrm{MUL}^a_{n-1,n}) \geq B + 1$, then $\pi - \mathrm{OBDD}(\mathrm{MUL}_{n-1,n}) \geq 2B$. This is, because in any OBDD computing $\mathrm{MUL}_{n-1,n}(x, y)$ either the input x or the input y may be set to the constant a. In both cases the resulting OBDD contains at at least $B - 1$ inner nodes, not counting those for variables fixed to constants (since they may be deleted without changing the function). So, by the last theorem the OBDD for $\mathrm{MUL}_{n-1,n}$ has a size of at least $2 \cdot 2^{\lfloor n/2 \rfloor}/192$, which proves the main result (Theorem 1).

Furthermore, by a straightforward reduction, one can easily obtain a lower bound on computing the other output bits of the multiplication. A simple proof (see [8], Corollary 1) shows, that any representation computing $\mathrm{MUL}_{k-1,n}$ or $\mathrm{MUL}_{2n-k-1,n}$ may also compute $\mathrm{MUL}_{k-1,k}$.

Corollary 1 *The size of an OBDD computing* $\mathrm{MUL}_{k-1,n}$ *or* MUL_{2n-k-1} *is at least* $2^{\lfloor k/2 \rfloor}/96$.

Note, that our lower bound on $\mathrm{MUL}_{n-1,n}$ relies only on the existence of a constant a for each variable ordering π, for which $\mathrm{MUL}^a_{n-1,n}$ leads to a large π-OBDD representation. If one would want to significantly improve this bound, this would have to be done by a different technique, taking more values for a into consideration. In other words, the result of Theorem 4 is optimal up to a constant factor.

Theorem 5 *There exists a variable ordering π which allows for any $a \in \{0, \ldots, 2^n - 1\}$ the construction of a $\pi - OBDD$ for* $\mathrm{MUL}^a_{n-1,n}$ *having a size of* $3 \cdot 2^{\lceil n/2 \rceil}$.

The proof will be sketched at the end of the next section.

4 Upper Bounds

In this section, we derive the upper bounds stated in Theorems 2 and 5. Both bounds can be proven by the same technique, which makes use of the fact, that the minimal-size π-OBDD for a Boolean function f is unique up to isomorphism [7], and of a theorem by Sieling and Wegener [19], describing the structure of the minimal-size π-OBDD.

Let f be a Boolean function and π be an arbitrary variable ordering on X_n. For $a_1, \ldots, a_i \in \{0,1\}$ $(1 \leq i \leq n)$, denote by $f_{a_1,\ldots,a_i}$ the subfunction of f that computes $f(x_1, \ldots, x_n)$, where for $1 \leq j \leq i$ the j-th input-variable according to π (that is $x_{\pi^{-1}(j)}$) is fixed by the constant a_j. More formally,

$$f_{a_1,\ldots,a_i} \;:=\; f_{|x_{\pi^{-1}(1)}=a_1,\ldots,x_{\pi^{-1}(i)}=a_i}.$$

Further, we say that a function g essentially depends on an input variable x_i, if $g_{|x_i=0} \neq g_{|x_i=1}$.

Theorem 6 ([19]) *The number of x_i-nodes of the minimal-size π-OBDD for f is the number of different subfunctions $f_{a_1,\ldots,a_{i-1}}$ for $a_1, \ldots, a_{i-1} \in \{0,1\}$, essentially depending on x_i.*

In order to show the stated upper bounds, let $x = (x_{n-1} \ldots x_0)$ and $y = (y_{n-1} \ldots y_0)$ be the input variables for $\mathrm{MUL}_{n-1,n}$. Further, let $\mathcal{F}_i$ denote the family of subfunctions f_{x,y^*} of $\mathrm{MUL}_{n-1,n}$ that result from replacing the variables $x_0, \ldots, x_{n-1}$ and $y_0, \ldots, y_{i-1}$ with constants. I.e., for $y^* := y_{i-1} \ldots y_0$,

$$f_{x,y^*}(y_{n-1} \ldots y_i) \;=\; \mathrm{MUL}_{n-1,n}\big(x,\, y_{n-1} \ldots y_i\, y^*\big).$$

Our goal is to bound the number of different subfunctions in $\mathcal{F}_i$. We define for any subfunction $f_{x,y^*} \in \mathcal{F}_i$ its index $ind(f_{x,y*})$ to be the number represented by $(z_{n-1} \ldots z_i)$, where $z = x \cdot y^*$. Consider arbitrary x and $y = (y_{n-1} \ldots y_i\, y^*)$. By the school-method of multiplication we have

$$x \cdot y \;=\; x \cdot y^* + 2^i x \cdot (y_{n-1} \ldots y_i).$$

Since the second term of the sum is a value shifted by i bits to the left (and thus has its i least significant bits set to 0), the addition of xy^* and $2^i x \cdot (y_{n-1} \ldots y_i)$ has no carry at position i. Hence, replacing $x \cdot y^*$ by $2^i \cdot ind(f_{x,y^*})$ in the above sum does not change the result for the output bits with indices $i, \ldots, n-1$. Furthermore, writing $2^i x \cdot (y_{n-1} \ldots y_i)$ as

$$\sum_{j=0}^{n-1} 2^{i+j} x_j \cdot (y_{n-1} \ldots y_i),$$

implies that the bits x_j with $j \geq n-i$ have no influence on the output bit with index $n-1$. Thus, $\mathrm{MUL}_{n-1,n}(x,y)$ is uniquely determined by $ind(f_{x,y^*})$ and $2^i(x_{n-i-1}\ldots x_0)\cdot(y_{n-1}\ldots y_i)$. We summarize this result in the following claim:

Claim 1 Each subfunction $f_{x,y^*} \in \mathcal{F}_i$ is uniquely determined by $(x_{n-i-1}\ldots x_0)$ and its index $ind(f_{x,y^*})$. ∎

We are now ready to proof the upper bounds.

Proof to Theorem 2: Let G be the minimal-size OBDD, which reads first all x-bits and then the bits $y_0,\ldots,y_{n-1}$ in this order. Further, let $k = \lceil n/3 \rceil$. Denote the upper part of G to the subgraph, in which the x-variables and the variables $y_0,\ldots,y_{k-1}$ are read. Obviously, this part contains at most as many nodes as a balanced binary tree with $n+k$ levels, thus has a size of at most $2^{n+k}-1$.

We bound now the number of y_i-nodes, for $i \geq k$. By Theorem 6, this is at most the number of different subfunctions f_{x,y^*} in $\mathcal{F}_i$. But since there are only 2^{n-i} different values for $ind(f_{x,y^*})$ and as many values for $(x_{n-i-1}\ldots x_0)$, it follows from Claim 1, that there are at most $2^{2(n-i)}$ different subfunctions in $\mathcal{F}_i$. So, the bottom part of G consists of $2^{2(n-i)}$ inner nodes for each $k \leq i \leq n-1$. An easy calculation shows, that both parts contain together

$$2^{n+k}-1+\sum_{i=k}^{n-1} 2^{2(n-i)} \;=\; 2^{n+k}+\frac{4}{3}\cdot 2^{2n-2k}-\frac{7}{3}$$

inner nodes. Since $k=\lceil n/3\rceil$, we may write $n=3k-\tau$ for some $0\leq\tau<3$. Thus, also counting the two sinks, the π-OBDD-size is bounded by

$$2^{4k-\tau}+\frac{4}{3}\cdot 2^{4k-2\tau}-\frac{7}{3}+2 \;\leq\; 2^{4k-4\tau/3}\left(2^{\tau/3}+\frac{4}{3}\cdot 2^{-2\tau/3}\right).$$

By case distinction ($\tau = 0,1,2$) it can be easily verified, that the factor in parenthesis has a value of at most $7/3$. Since further the exponent of the first factor $(4k-4\tau/3)$ equals $4n/3$, the proof is complete. ∎

The proof of Theorem 5 for $\mathrm{MUL}^a_{n-1,n}$ uses almost the same line of argument, so that we only sketch the differences. The vector x of variables is replaced with the constant a, and the variables $y_0,\ldots,y_{n-1}$ are read again in this order. But now the upper part consists of the first $n/2$ variables of y, that is $y_0,\ldots,y_{n/2-1}$, and is again bounded by the size of a binary tree $(2^{n/2}-1)$. Using the index of the functions f_{a,y^*}, the number of different subfunctions in $\mathcal{F}_i$ is then bounded for $n/2 \leq i \leq n-1$ similarly to the above proof. In this way, we conclude that the lower part of the OBDD consists of at most $\sum_{i=n/2}^{n-1} 2^{n-i} = 2\left(2^{n/2}-1\right)$ inner nodes, which shows the claim.

Acknowledgments

I would like to thank Ingo Wegener for the valuable hints leading to several improvements, as well as Beate Bollig for her helpful comments.

References

[1] S. B. Akers. Binary decision diagrams. *IEEE Transactions on Computers*, C-27:509–516, 1978.

[2] M. Atici and D. R. Stinson. Universal hashing and multiple authentication. In *Advances in Cryptology – CRYPTO '96*, pp. 16–30. 1996.

[3] P. Beame, M. Tompa, and P. Yan. Communication-space tradeoffs for unrestricted protocols. *SIAM Journal on Computing*, 23:652–661, 1994.

[4] B. Bollig and I. Wegener. Improving the variable ordering of OBDDs is NP-complete. *IEEE Transactions on Computers*, 45:993–1002, 1996.

[5] B. Bollig and I. Wegener. Asymptotically optimal bounds for OBDDs and the solution of some basic OBDD problems. In *Proceedings of the 25th International Colloquium on Automata, Languages, and Programming.* 2000. To Appear.

[6] Y. Breibart, H. B. Hunt III, and D. Rosenkrantz. On the size of binary decision diagrams representing Boolean functions. *Theoretical Computer Science*, 145:45–69, 1995.

[7] R. E. Bryant. Graph-based algorithms for boolean function manipulation. *IEEE Transactions on Computers*, C-35:677–691, 1986.

[8] R. E. Bryant. On the complexity of VLSI implementations and graph representations of boolean functions with applications to integer multiplication. *IEEE Transactions on Computers*, 40:205–213, 1991.

[9] J. L. Carter and M. N. Wegman. Universal classes of hash functions. *Journal of Computer and System Sciences*, 18:143–154, 1979.

[10] M. Dietzfelbinger. Universal hashing and k-wise independent random variables via integer arithmetic without primes. In *Proceedings of the 13th Annual Symposium on Theoretical Aspects of Computer Science*, pp. 569–580. 1996.

[11] M. Dietzfelbinger, T. Hagerup, J. Katajainen, and M. Penttonen. A reliable randomized algorithm for the closest-pair problem. *Journal of Algorithms*, 25:19–51, 1997.

[12] M. Dietzfelbinger, A. Karlin, K. Mehlhorn, F. Meyer auf der Heide, H. Rohnert, and R. E. Tarjan. Dynamic perfect hashing: Upper and lower bounds. *SIAM Journal on Computing*, 23:738–761, 1994.

[13] M. L. Fredman, J. Komlós, and E. Szemerédi. Storing a sparse table with $O(1)$ worst case access time. *Journal of the Association for Computing Machinery*, 31:538–544, 1984.

[14] R. Impagliazzo and D. Zuckerman. How to recycle random bits. In *Proceedings of the 30th Annual Symposium on Foundations of Computer Science*, pp. 248–253. 1989.

[15] C. Y. Lee. Representation of switching circuits by binary-decision programs. *The Bell Systems Technical Journal*, 38:985–999, 1959.

[16] Y. Mansour, N. Nisan, and P. Tiwari. The computational complexity of universal hashing. *Theoretical Computer Science*, 107:121–133, 1993.

[17] P. Rogaway. Bucket hashing and its application to fast message authentication. In *Advances in Cryptology – CRYPTO '95*, pp. 29–42. 1995.

[18] D. Sieling. On the existence of polynomial time approximation schemes for OBDD minimization. In *Proceedings of the 15th Annual Symposium on Theoretical Aspects of Computer Science*, pp. 205–215. 1998.

[19] D. Sieling and I. Wegener. NC-algorithms for operations on binary decision diagrams. *Parallel Processing Letters*, 48:139–144, 1993.

[20] M. Sipser. A complexity theoretic approach to randomness. In *Proceedings of the 15th Annual ACM Symposium on Theory of Computing*, pp. 330–335. 1983.

[21] I. Wegener. *Branching Programs and Binary Decision Diagrams - Theory and Applications*. SIAM Monographs in Discrete Mathematics and Its Applications. In print, 2000.

[22] M. N. Wegman and J. L. Carter. New classes and applications of hash functions. In *Proceedings of the 20th Annual IEEE Symposium on Foundations of Computer Science*, pp. 175–182. 1979.

[23] P. Woelfel. Efficient strongly universal and optimally universal hashing. In *Mathematical Foundations of Computer Science 1999: 24th International Symposium*, pp. 262–272. 1999.

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'