Preprint from Proceedings of ICCAD'98, San Jose, California, November 8-12, 1998

Word-Level Decision Diagrams, WLCDs and Division

Christoph Scholl   Bernd Becker   Thomas M. Weis

Institute of Computer Science
Albert–Ludwigs–University
D 79110 Freiburg im Breisgau, Germany
email: <name>@informatik.uni-freiburg.de

Abstract

Several types of Decision Diagrams (DDs) have been proposed for the verification of Integrated Circuits. Recently, word-level DDs like BMD*s,* *BMD*s,* HDD*s,* K*BMD*s and* *PHDD*s have been attracting more and more interest, e.g., by using* *BMD*s and* *PHDD*s it was for the first time possible to formally verify integer multipliers and floating point multipliers of "significant" bitlengths, respectively.*

On the other hand, it has been unknown, whether division*, the operation inverse to multiplication, can be efficiently represented by some type of word-level DDs. In this paper we show that the representational power of any word-level DD is too weak to efficiently represent integer division. Thus, neither a clever choice of the variable ordering, the decomposition type or the edge weights, can lead to a polynomial DD size for division.*

For the proof we introduce Word-Level Linear Combination Diagrams (WLCDs)*, a DD, which may be viewed as a "generic" word-level DD. We derive an exponential lower bound on the* WLCD *representation size for integer dividers and show how this bound transfers to all other word-level DDs.*

1 Introduction

One of the most important tasks during the design of *Integrated Circuits* is the verification of an implemented circuit, i.e., the check whether the implementation fulfills its specification.

In the last few years several methods based on *Decision Diagrams* (DDs) have been proposed [14, 5] to perform verification. The idea is to transform both, implementation and specification of a combinational circuit, into a DD. Then, due to the canonicity of the DD representation, the equivalence check for specification and implementation translates to the check whether the corresponding DDs are identical.

The most popular data structure in this context are *Ordered Binary Decision Diagrams* (OBDDs) [3]. They were applied successfully e.g. to the verification of control logic and integer adders. But there are functions of high practical relevance, which cannot be represented efficiently by OBDDs. Already in [3] and [4] Bryant proved that OBDD representations for integer multipliers are of exponential size.

Several other types of DDs were defined to overcome the limitations of OBDDs, such as *Ordered Functional Decision Diagrams* (OFDDs) [12], *Ordered Kronecker Functional Decision Diagrams* (OKFDDs) [11], *Multi–Terminal Binary Decision Diagrams* (MTBDDs) [9, 1] and *Edge–valued Binary Decision Diagrams* (EVBDDs) [13]. But the first DDs to represent integer multiplication efficiently were *Binary Moment Diagrams* (BMDs) and *Multiplicative* BMD*s* (*BMDs) introduced in [6]. Like MTBDDs and EVBDDs, also BMDs and *BMDs are word-level DDs, i.e. they represent integer-valued functions $f : \{0,1\}^n \to \mathbb{Z}$.

To further improve on the representational power of BMDs, several other word-level DD types have been introduced, e.g. *Hybrid Decision Diagrams* (HDDs) [8] and *Kronecker* *BMD*s* (K*BMDs) [10]. Recently Chen and Bryant defined a new data structure called *Multiplicative Power Hybrid Decision Diagrams* (*PHDDs) [7], which is able to represent not only integer multiplication but also floating point multiplication efficiently.

Until now it was not known, whether the word-level DDs mentioned above are also able to represent division efficiently. Recently Nakanishi [15] made a first step by showing that *BMDs cannot represent integer division efficiently. The proof is technically complicated, it is based on fooling set arguments similar to the orginal proof for multiplication by Bryant and has to take into account the edge values in the *BMD representation. Consequently, as already mentioned, in this form it only works for *BMDs.

In this paper we prove that integer division cannot be represented in polynomial size *by any of the ordered word-level DDs mentioned in the literature until now*. Even more interestingly, we prove that the *concept of word-level DDs in general is too weak to result in polynomial size representations of division.*

For the proof we introduce a new data structure, the *Word-Level Linear Combination Diagrams* (WLCDs). WLCDs are a generalization of Waack's *Parity Ordered Binary Decision Diagrams* (POBDDs) [17] to the word level. It turns out that WLCDs can be viewed as a "generic" ordered word-level DD in the following sense: Each ordered word-level DD can be "embedded into" WLCDs such that a DD with k nodes is transformed into a WLCD representing the same function with the same number k of nodes. Thus, a lower bound on the size of a WLCD is also a lower bound on the size of all other ordered word-level DDs.

We apply this idea to integer division by deriving an exponential lower bound on the size of WLCDs representing integer divison (regardless of the chosen variable order). For WLCDs lower bounds can be obtained by consideration of the rank of a communication matrix which is constructed from the function tables of several cofactors. It follows that bothering details concerning e.g. edge values have not to be taken into account to derive the lower bound in our proof. On the other hand, due to the properties of WLCDs we obtain an exponential lower bound result, valid for all ordered word-level DD types.

The paper is structured as follows. In Section 2 we provide basics on word-level DDs which will be necessary for the understanding of the paper. WLCDs and their relationship to existing word-level

DDs are introduced in Section 3. Furthermore, an algebraic characterization of the WLCD complexity is given which leads to the rank considerations of certain cofactor matrices. In Section 4 the lower bound for division is derived [1]. We finish with conclusions and perspectives of further work in Section 5.

2 Preliminaries: Word-Level Decision Diagrams

In this section we give a short review of ordered word-level DDs, data structures used for the representation of so-called *Pseudo Boolean* functions, i.e. functions from a Boolean domain to the integers or rational numbers. In general, DDs are graph–based representations, where at each (non–terminal) node (labeled with a variable x) a decomposition of the function (represented by this node) into two subfunctions (the *low*–function and the *high*–function) is performed:

Definition 1 *A word-level DD is a rooted directed acyclic graph $G = (V, E)$ with non empty vertex set V containing two types of vertices,* non-terminal *and* terminal *vertices. A non-terminal vertex v has as label a variable $index(v) \in \{x_1, \ldots, x_n\}$ and two children $low(v), high(v) \in V$. A terminal vertex v is labeled with a value $value(v) \in \mathbb{Z}$.*

For the purpose of this paper, we are only interested in *ordered* DDs, i.e. DDs, where the variables occur in the same order on all paths of the DD. More precisely, this means:

Definition 2 *A DD is ordered iff there is a fixed order $\pi : \{1, \ldots, n\} \to \{x_1, \ldots, x_n\}$ such that for any non-terminal vertex v the following holds: $index(low(v)) = \pi(k)$ with $k > \pi^{-1}(index(v))$ ($index(high(v)) = \pi(q)$ with $q > \pi^{-1}(index(v))$) as long as $low(v)$ ($high(v)$) is also a non-terminal vertex.*

Based on these general definitions we now consider different decomposition types and shortly discuss resulting word-level DDs and corresponding evaluation rules. (For a survey on word-level DDs and more details see also [2].)

2.1 Decomposition types and evaluation rules

In word-level DDs the function $f_v : \{0,1\}^n \to \mathbb{Q}$ represented by a non–terminal node v, which is labeled by variable x_i, is decomposed into two subfunctions, both independent of variable x_i. Depending on the decomposition type these subfunctions are combined from the cofactors

$$(f_v)_{\overline{x_i}} = f_v(x_1, \ldots, x_{i-1}, 0, x_{i+1}, \ldots, x_n)$$

and

$$(f_v)_{x_i} = f_v(x_1, \ldots, x_{i-1}, 1, x_{i+1}, \ldots, x_n)$$

in different ways. DDs as defined in literature differ in the way they use decomposition types. Decomposition types can be defined by the set $\mathbb{Z}_{2,2}$ of non–singular 2×2 matrices over $\mathbb{Z}$ [8]. The most important decomposition types are *Shannon decomposition*, *positive Davio decomposition* and *negative Davio decomposition*. The Shannon decomposition is used in MTBDDs [9] and EVBDDs [13], the positive Davio decomposition is used in BMDs and *BMDs [6]. In K*BMDs [10] and *PHDDs [7] Shannon decomposition, positive Davio and negative Davio decomposition are used. In HDDs [8] six different decomposition types (including Shannon, positive and negative Davio decomposition) are used.

Following [8] the matrices corresponding to Shannon, positive Davio and negative Davio decomposition, respectively, are

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \qquad \begin{pmatrix} 1 & 0 \\ -1 & 1 \end{pmatrix} \quad \text{and} \quad \begin{pmatrix} 0 & 1 \\ 1 & -1 \end{pmatrix}.$$

[1] Due to lack of space some details of the proof are omitted. They can be found in [16].

These matrices define how the functions $f_{low(v)}$ and $f_{high(v)}$ represented by $low(v)$ and $high(v)$ are computed from $(f_v)_{\overline{x_i}}$ and $(f_v)_{x_i}$. For the positive Davio decomposition, e.g., we have

$$\begin{pmatrix} f_{low(v)} \\ f_{high(v)} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ -1 & 1 \end{pmatrix} \begin{pmatrix} (f_v)_{\overline{x_i}} \\ (f_v)_{x_i} \end{pmatrix},$$

i.e., $f_{low(v)} = (f_v)_{\overline{x_i}}$ and $f_{high(v)} = (f_v)_{x_i} - (f_v)_{\overline{x_i}}$.

A terminal node v with $value(v) = z$ represents the constant function with function value z. To evaluate the function f_v represented by a non–terminal node v for $x_i = 0$ or $x_i = 1$, we have to reconstruct $(f_v)_{\overline{x_i}}$ or $(f_v)_{x_i}$ from $f_{low(v)}$ and $f_{high(v)}$. To do so, we make use of the fact, that the decomposition type matrices are non–singular: Since a decomposition type matrix A is non–singular, the inverse matrix A^{-1} exists and

$$\begin{pmatrix} (f_v)_{\overline{x_i}} \\ (f_v)_{x_i} \end{pmatrix} = A^{-1} \cdot \begin{pmatrix} f_{low(v)} \\ f_{high(v)} \end{pmatrix}. \tag{1}$$

The inverse decomposition type matrices for Shannon, positive Davio and negative Davio decomposition, respectively, are

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \qquad \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} \quad \text{and} \quad \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}.$$

For positive Davio decomposition, e.g., this means that $(f_v)_{\overline{x_i}} = f_{low(v)}$ and $(f_v)_{x_i} = f_{low(v)} + f_{high(v)}$.

2.2 Additive edge values, multiplicative edge values, negation edges

Edge values are introduced to increase the amount of subgraph sharing when using integer–valued terminal nodes. It has to be differentiated between *additive* and *multiplicative* edge values.

An edge with additive weight a and multiplicative weight m leading to node v represents the function

$$< (a, m), f_v > := a + m \cdot f_v. \tag{2}$$

MTBDDs, BMDs and HDDs use no edge values, EVBDDs use only additive weights, i.e., the multiplicative weight m is 1, *BMDs use only multiplicative weights, i.e. $a = 0$. K*BMDs use both additive and multiplicative weights. *PHDDs use only multiplicative weights of form $(-1)^{ne} \cdot 2^w$ with $ne \in \{0,1\}$ and $w \in \mathbb{Z}$. (For reasons of memory efficiency $(-1)^{ne} \cdot 2^w$ is stored as an integer w and a bit ne representing a "negation edge" when $ne = 1$.)

Now consider any ordered word-level DD with edge values. Then for each non–terminal node v there is a 0–edge labeled with edge weights (a_{low}, m_{low}) leading to node $low(v)$ and a 1–edge labeled with edge weights (a_{high}, m_{high}) leading to node $high(v)$. If in node v the decomposition type $A = \binom{a_{11}\ a_{12}}{a_{21}\ a_{22}}$ with inverse matrix $A^{-1} = \binom{a'_{11}\ a'_{12}}{a'_{21}\ a'_{22}}$ is used, then using Equations 1 and 2 the evaluation rule for this node is the following:

$$\begin{aligned}
f_v &= (1 - x_i)\cdot(f_v)_{\overline{x_i}} + x_i \cdot (f_v)_{x_i} \\
&= (1 - x_i)\cdot(a'_{11}(a_{low} + m_{low} f_{low(v)}) \\
&\qquad\qquad + a'_{12}(a_{high} + m_{high} f_{high(v)})) \\
&\quad + x_i \cdot \quad (a'_{21}(a_{low} + m_{low} f_{low(v)}) \\
&\qquad\qquad + a'_{22}(a_{high} + m_{high} f_{high(v)})) \\
&= (1 - x_i)\cdot((a'_{11} a_{low} + a'_{12} a_{high}) \\
&\qquad\qquad + (a'_{11} m_{low} f_{low(v)}) + (a'_{12} m_{high} f_{high(v)})) \\
&\quad + x_i \cdot \quad ((a'_{21} a_{low} + a'_{22} a_{high}) \\
&\qquad\qquad + (a'_{21} m_{low} f_{low(v)}) + (a'_{22} m_{high} f_{high(v)})).
\end{aligned} \tag{3}$$

In Section 3 we will use the "most general evaluation rule" of Equation 3 to analyze the relationship between the existing ordered word-level DDs and our new data structure called Word-Level Linear Combination Diagrams (WLCDs).

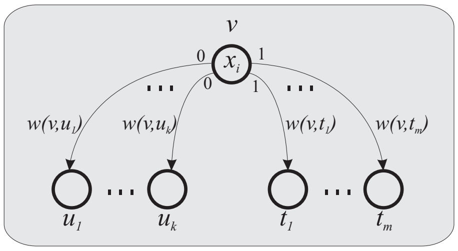

Figure 1: Non-terminal vertex v of a WLCD. v is labeled by variable x_i. The 0–edges of v are given by edges to nodes $u_1, \ldots, u_k$ and the 1–edges are given by edges to $t_1, \ldots, t_m$.

3 Word-Level Linear Combination Diagrams

In this section we define Word-Level Linear Combination Diagrams (WLCDs). WLCDs are a generalization of POBDDs defined by Waack [17] to the word-level. Whereas POBDDs can represent only Boolean functions, WLCDs represent functions $f : \{0,1\}^n \to \mathbb{Q}$.

WLCDs are given by the following definition:

Definition 3 *A Word-Level Linear Combination Diagram (*WLCD*) is a rooted directed acyclic graph $G = (V, E)$. If the* WLCD *is not empty, it contains exactly one sink labeled with 1 and with no outgoing edges. The remaining nodes are called non–terminal nodes. A non-terminal vertex v is labeled by a variable $index(v) \in \{x_1, \ldots, x_n\}$. The outgoing edges of a non–terminal node v are partitioned into two sets: 0–edges(v) and 1–edges(v). At least one of these sets is not empty. All edges e are labeled by an edge weight $w(e) \in \mathbb{Q}$. A* WLCD *is ordered, i.e., as with DDs the variables occur in the same order on all paths of* WLCD. *The size of a* WLCD *is its number of nodes.*

The definition of a WLCD is illustrated by Figure 1.

An empty WLCD represents the constant 0–function, the sink of a non–empty WLCD represents the constant 1–function. The function f_v represented by a non–terminal node v labeled by variable x_i with *0–edges(v)* $= \{(v,u_1), \ldots, (v,u_k)\}$ and *1–edges(v)* $= \{(v,t_1), \ldots, (v,t_m)\}$ is defined by the following evaluation rule:

$$\begin{aligned} f_v \quad := \quad & (1-x_i)\cdot(w(v,u_1)\cdot f_{u_1} + \ldots + w(v,u_k)\cdot f_{u_k}) \\ & + x_i \cdot (w(v,t_1)\cdot f_{t_1} + \ldots + w(v,t_m)\cdot f_{t_m}). \end{aligned} \tag{4}$$

Similar to POBDDs, also for WLCDs efficient synthesis operations and an equivalence check can be derived. We omit any further details, rather we concentrate on the property of WLCDs which is most important in this paper: Ordered word-level DDs can be "embedded into WLCDs", i.e., if there is some word-level DD with *k* nodes, we can easily construct a WLCD with the same number *k* of nodes. This fact is used to conclude lower bounds on the size of arbitrary word-level DDs from lower bounds on the size of WLCDs.

The computation of lower bounds on the size of WLCDs can be done in an elegant way using arguments from linear algebra. Before coming to lower bounds we show how to embed word-level DDs in WLCDs.

3.1 Relationship between WLCDs and existing word-level DDs

Here we prove that all ordered word-level DDs mentioned in the previous sections can be "embedded into WLCDs". To do so we proceed as follows:

A given word-level DD is transformed step by step into a WLCD.

If the given DD contains terminals v with values $value(v)$ different from 0 and 1, these terminals are replaced by a terminal 1 and the multiplictive edge weights of all incoming edges of v are multiplied by $value(v)$. If now there is more than one terminal with value 1, these terminals are replaced by a unique terminal with value 1. Edges to terminal 0 with additive weight $a \neq 0$ are replaced by edges to terminal 1 with additive weight a and multiplicative weight 0. The 0–terminal is removed. All these steps do not change the function represented by the DD.

Now in a bottom–up procedure for each non–terminal node v labeled with variable $x_i = index(v)$ representing a function f_v the outgoing edges are replaced resulting in a WLCD–node representing the same function f_v. Suppose that the decomposition type used for node v is given by $A = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}$ (with inverse matrix $A^{-1} = \begin{pmatrix} a'_{11} & a'_{12} \\ a'_{21} & a'_{22} \end{pmatrix}$) and the 0–edge is labeled with edge weights (a_{low}, m_{low}), the 1–edge is labeled with edge weights (a_{high}, m_{high}). Then the evaluation rule of Equation 3 gives a relation between $f_{low(v)}$ and $f_{high(v)}$ and f_v. A comparison with the evaluation rule for WLCDs (see Equation 4) leads to the definition of the equivalent WLCD–node and its corresponding edges (let v_{one} be the terminal with value 1):

- *0–edges(v)* $= \{(v, v_{one}), (v, low(v)), (v, high(v))\}$,
  $w(v, v_{one}) = a'_{11} a_{low} + a'_{12} a_{high}$,
  $w(v, low(v)) = a'_{11} m_{low}$,
  $w(v, high(v)) = a'_{12} m_{high}$.
- *1–edges(v)* $= \{(v, v_{one}), (v, low(v)), (v, high(v))\}$,
  $w(v, v_{one}) = a'_{21} a_{low} + a'_{22} a_{high}$,
  $w(v, low(v)) = a'_{21} m_{low}$,
  $w(v, high(v)) = a'_{22} m_{high}$.

The replacement is illustrated by Figure 2.

After this bottom–up procedure, if there is a root edge with weight (a, m), the weights of the outgoing edges of the root are multiplied by m and an edge $(root, v_{one})$ with weight a is included into *0–edges(root)* and *1–edges(root)*.

Finally we obtain a WLCD representing the same function as the original DD. We summarize:

Theorem 1 *If the* MTBDD, EVBDD, BMD, *BMD, HDD, K*BMD *or* *PHDD *for a function $f : \{0,1\}^n \to \mathbb{Z}$ (or $f : \{0,1\}^n \to \mathbb{Q}$ for the case of* *PHDDs*) with variable order π has k nodes, then there also exists a* WLCD *with variable order π representing f with (at most) k nodes.*

Example 1 *In Figure 3 the node replacement described to prove Theorem 1 is illustrated for positive Davio decomposition without edge weights (i.e. the additive edge weights are* 0 *and multiplicative edge weights are* 1*). For positive Davio decomposition the decomposition type matrix ist given by $A = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ -1 & 1 \end{pmatrix}$, $A^{-1} = \begin{pmatrix} a'_{11} & a'_{12} \\ a'_{21} & a'_{22} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}$. Thus, the evaluation rule can be simplified to*

$$\begin{aligned} f_v \quad = \quad & (1-x_i)\cdot((a'_{11}a_{low} + a'_{12}a_{high}) \\ & \qquad + (a'_{11} m_{low} f_{low(v)}) + (a'_{12} m_{high} f_{high(v)})) \\ & + x_i \cdot \quad ((a'_{21}a_{low} + a'_{22}a_{high}) \\ & \qquad + (a'_{21} m_{low} f_{low(v)}) + (a'_{22} m_{high} f_{high(v)})) \\ = \quad & (1-x_i)\cdot f_{low(v)} + x_i \cdot (f_{low(v)} + f_{high(v)}). \end{aligned}$$

3.2 An Algebraic Characterization of the WLCD Complexity

In this subsection we give an algebraic characterization of the WLCD complexity, which we will use to prove lower bounds on

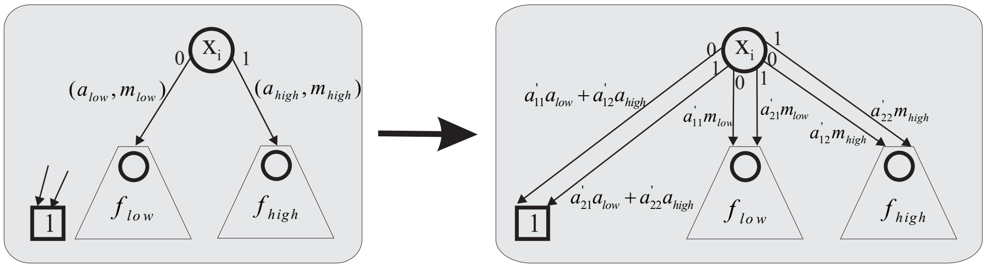

Figure 2: Transformation into WLCD node

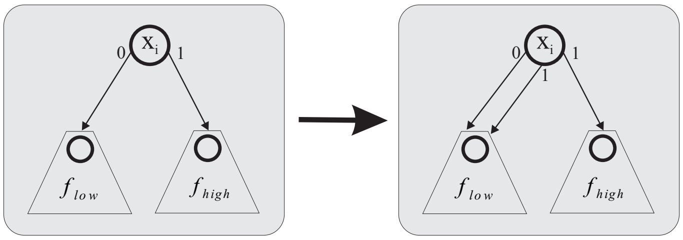

Figure 3: Transformation of a positive Davio node into a WLCD node

the size of WLCDs. We show, that the number of nodes in a WLCD cannot be smaller than the dimension of a certain vector space.

Consider the set of all functions from $\{0,1\}^n$ to the rational numbers $Map(\{0,1\}^n, \mathbb{Q}) = \{f : \{0,1\}^n \to \mathbb{Q}\}$. Define addition on $Map(\{0,1\}^n, \mathbb{Q})$ by $(f+g)(x_1,\ldots,x_n) = f(x_1,\ldots,x_n) + g(x_1,\ldots,x_n)$ and multiplication with a scalar $w \in \mathbb{Q}$ by $(w \cdot f)(x_1,\ldots,x_n) = w \cdot f(x_1,\ldots,x_n)$. It is easy to see, that $Map(\{0,1\}^n, \mathbb{Q})$ together with addition and multiplication with scalars from $\mathbb{Q}$ forms a vector space.

Based on WLCDs with fixed variable order π we will define subspaces of the vector space $Map(\{0,1\}^n, \mathbb{Q})$. W.l.o.g. we assume the *natural* variable order, i.e., $\pi : \{1,\ldots,n\} \to \{x_1,\ldots,x_n\}$ with $\pi(i) = x_i \; \forall i \in \{1,\ldots,n\}$.

Given a WLCD $\mathcal{B}$, consider for some $k \in \{1,\ldots,n\}$ the set of all WLCD–nodes, which are labeled with variable x_k or which are labeled with a variable x_i with $i > k$ and which have an incoming edge from a node labeled by a variable x_j with $j < k$. These nodes represent functions of $Map(\{0,1\}^n, \mathbb{Q})$. We denote this set of functions by $V_k^{\mathcal{B}}$. Of course, the vector space $< V_k^{\mathcal{B}} >$ which is generated by the functions in $V_k^{\mathcal{B}}$ forms a subspace of $Map(\{0,1\}^n, \mathbb{Q})$.

Let f be the function represented by the WLCD $\mathcal{B}$. We consider the following set of cofactors of f:

$$V_k^f = \{f|_{x_1=c_1,\ldots,x_{k-1}=c_{k-1}} | c_1,\ldots,c_{k-1} \in \{0,1\}\}.$$

Again, $< V_k^f >$, which is generated by the functions in V_k^f, is a subspace of $Map(\{0,1\}^n, \mathbb{Q})$.

Now we investigate the relationship between the vector spaces $< V_k^{\mathcal{B}} >$ and $< V_k^f >$. We claim that

$$< V_k^f > \subseteq < V_k^{\mathcal{B}} > .$$

To prove this it is sufficient to show, that each cofactor $f|_{x_1=c_1,\ldots,x_{k-1}=c_{k-1}} \in V_k^f$ is in $< V_k^{\mathcal{B}} >$. We consider all paths starting from the root of $\mathcal{B}$, which fulfill the assignment $x_1 = c_1, \ldots, x_{k-1} = c_{k-1}$. Let $v_1, \ldots, v_m$ be the nodes, which are reached by these paths and suppose that each node $v_r \; \forall r \in \{1,\ldots, m\}$ is reached by i_r different paths $p_1^{(r)}, \ldots, p_{i_r}^{(r)}$. Let $w_j^{(r)}$ be the product of all weights of edges on path $p_j^{(r)}$. Then according to the definition of WLCDs and by induction on k the following holds:

$$f|_{x_1=c_1,\ldots,x_{k-1}=c_{k-1}} = \\ = \left(\sum_{j=1}^{i_1} w_j^{(1)}\right) f_{v_1} + \ldots + \left(\sum_{j=1}^{i_m} w_j^{(m)}\right) f_{v_m} .$$

Since $\forall 1 \le i \le m \; f_{v_i} \in V_k^{\mathcal{B}}$, we conclude that $f|_{x_1=c_1,\ldots,x_{k-1}=c_{k-1}} \in < V_k^{\mathcal{B}} >$ for each choice of $c_1, \ldots, c_{k-1} \in \{0,1\}$.

Because of $< V_k^f > \subseteq < V_k^{\mathcal{B}} >$ we have

$$dim(< V_k^f >) \le dim(< V_k^{\mathcal{B}} >)$$

and since $V_k^{\mathcal{B}}$ generates $< V_k^{\mathcal{B}} >$ it holds

$$dim(< V_k^f >) \le |V_k^{\mathcal{B}}|.$$

Thus we obtain the following lemma

Lemma 1 *Let f be any function in $Map(\{0,1\}^n, \mathbb{Q})$. Then*

$$dim(< V_k^f >)$$

is a lower bound on the size of a WLCD *for f with respect to the natural variable ordering.*

In fact, we can prove even a stronger result with similar arguments as in the proof of Waack [17] for POBDDs:

Theorem 2 *A* WLCD *$\mathcal{B}$ with natural variable order representing function f with a minimal number of nodes, has exactly $dim(< \bigcup_{k=1}^{n+1} V_k^f >)$ nodes.*

However, for the purposes of this paper we need only Lemma 1.

a_1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
a_0	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
b_1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
b_0	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
$a_3 a_2 b_3 b_2$																
0 0 0 0	*	0	0	0	*	1	0	0	*	2	1	0	*	3	1	1
0 0 0 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0 0 1 0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0 0 1 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0 1 0 0	*	4	2	1	*	5	2	1	*	6	3	2	*	7	3	2
0 1 0 1	1	0	0	0	1	1	0	0	1	1	1	0	1	1	1	1
0 1 1 0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0 1 1 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1 0 0 0	*	8	4	2	*	9	4	3	*	10	5	3	*	11	5	3
1 0 0 1	2	1	1	1	2	1	1	1	2	2	1	1	2	2	1	1
1 0 1 0	1	0	0	0	1	1	0	0	1	1	1	0	1	1	1	1
1 0 1 1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1 1 0 0	*	12	6	4	*	13	6	4	*	14	7	4	*	15	7	5
1 1 0 1	3	2	2	1	3	2	2	1	3	2	2	2	3	3	2	2
1 1 1 0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1 1 1 1	1	0	0	0	1	1	0	0	1	1	1	0	1	1	1	1

Figure 4: Communication matrix of division for $n = 4$.

4 An Exponential Lower Bound for Division

In this section we apply Lemma 1 to derive an exponential lower bound on the size of WLCDs (and thus of word-level DDs) representing integer divison.

For our proof we use the following notations and definitions concerning division: Two sets of variables, the a-variables $A = \{a_{n-1}, \ldots, a_0\}$ and the b-variables $B = \{b_{n-1}, \ldots, b_0\}$, are considered. As usual, the binary representation of A and B is given by

$$||A|| := 2^{n-1}a_{n-1} + \ldots + 2^0 a^0 \quad \text{and}$$
$$||B|| := 2^{n-1}b_{n-1} + \ldots + 2^0 b^0,$$

respectively. Then the integer division DIV is the Pseudo Boolean function defined by

$$\begin{aligned} DIV : \{0,1\}^n \times \{0,1\}^n &\rightarrow I\!N. \\ (a_{n-1}, \ldots, a_0, b_{n-1}, \ldots, b_0) &\mapsto \left\lfloor \frac{||A||}{||B||} \right\rfloor. \end{aligned}$$

Before we consider the general case for the proof of an exponential lower bound for WLCDs with arbitrary variable orders we have a look at a restricted case which nicely demonstrates the idea of the proof and the proof technique. The proof of the general case is slightly more complicated but works along similar lines. Due to lack of space we only give the idea of the proof for the general case. More details can be found in [16].

4.1 WLCDs with Interleaved Variable Ordering

For the restricted case we fix the variable order in advance: it is given by the *interleaved ordering* $(a_{n-1}, b_{n-1}, \ldots, a_0, b_0)$.

Furthermore, we may assume that n is even. (For n odd, we embed an $(n-1)$–bit divider into the n–bit divider by setting $a_{n-1} = b_{n-1} = 0$ and note that for an exponential lower bound $\Omega(c^n)$ it holds $\Omega(c^n) = \Omega(c^{n-1})$.

Following Lemma 1 we now consider the set V^f_{n+1} of cofactors $V^f_{n+1} = \{f|_{a_{n-1}=c_1, b_{n-1}=c_2, \ldots, a_{\frac{n}{2}}=c_{n-1}, b_{\frac{n}{2}}=c_n} \mid c_1, \ldots, c_n \in \{0,1\}\}$ and show an exponential lower bound for $dim(< V^f_{n+1} >)$.

To estimate $dim(< V^f_{n+1} >)$ we prove that a certain number of elements of V^f_{n+1} is linearly independent. For that we consider a *communication matrix* whose rows are function tables of the cofactors of V^f_{n+1}. The rows of the matrix are "numbered" by input combinations of the "upper half" of the a- and b-variables. Analogously, the "lower half" of the a- and b-variables defines the columns. For illustration see Figure 4, where we give the communication matrix for $n = 4$. (A star in the matrix means that the corresponding result of DIV is not defined (division by zero). Our proof is valid for all possible replacements of the stars.)

The rank of this communication matrix is equal to $dim(< V^f_{n+1} >)$. Since we need only a lower bound on $dim(< V^f_{n+1} >)$, we may remove columns and rows in the matrix (thereby possibly reducing the rank of the resulting matrix).

The idea now is to restrict to entries with constant values for the b-variables and to observe the result of the division for increasing values of a-variables . More precisely, we only keep rows where from the b-variables exactly the least significant upper b-variable $b_{\frac{n}{2}}$ is set, i.e. $b_{n-1} = 0, \ldots, b_{\frac{n}{2}+1} = 0, b_{\frac{n}{2}} = 1$.

Analogously, only columns with $b_{\frac{n}{2}-1} = 0, \ldots, b_1 = 0, b_0 = 1$ are considered. Furthermore, the rows and columns with 0 for all a-inputs are removed. For our example with $n = 4$ the following matrix remains:

a_1	0	1	1
a_0	1	0	1
b_1	0	0	0
b_0	1	1	1
$a_3 a_2 b_3 b_2$			
0 1 0 1	1	1	1
1 0 0 1	1	2	2
1 1 0 1	2	2	3

In general, we obtain a matrix M of size $(2^{\frac{n}{2}} - 1) \times (2^{\frac{n}{2}} - 1)$. For the computation of the entries of M consider the set $A_{HIGH} := \{a_{n-1}, \ldots, a_{\frac{n}{2}}\}$ and $A_{LOW} := \{a_{\frac{n}{2}-1}, \ldots, a_0\}$, i.e. A_{HIGH} (A_{LOW}) consists of the upper (lower) a-variables. Define $||A_{HIGH}|| := \sum_{i=\frac{n}{2}}^{n-1} a_i 2^{i-\frac{n}{2}}$ and $||A_{LOW}|| := \sum_{i=0}^{\frac{n}{2}-1} a_i 2^i$.

Now let m_{ij} denote an entry of M. Then row i corresponds to an assignment $||A_{HIGH}||$ and column j corresponds to an assignment $||A_{LOW}||$. The entry m_{ij} is then given by

$$m_{ij} = \left\lfloor \frac{||A_{HIGH}|| \cdot 2^{\frac{n}{2}} + ||A_{LOW}||}{2^{\frac{n}{2}} + 1} \right\rfloor.$$

(Remember that $(b_{n-1}, \ldots, b_{\frac{n}{2}}) = (0, \ldots, 0, 1)$ and $(b_{\frac{n}{2}-1}, \ldots, b_0) = (0, \ldots, 0, 1)$.)

It follows that the result of DIV cannot be determined by looking at the assignment for A_{HIGH}, rather the "relation" between "corresponding" bits of A_{HIGH} and A_{LOW} is essential: For $||A_{HIGH}|| = ||A_{LOW}||$ the result is obviously $||A_{HIGH}||$. For $||A_{LOW}|| < ||A_{HIGH}||$ we have $m_{ij} = ||A_{HIGH}|| - 1$.

($m_{ij} < ||A_{HIGH}|| - 1$ would imply $||A_{HIGH}|| \geq 2^{\frac{n}{2}} + 1 + ||A_{LOW}||$ which cannot be true.) For $||A_{LOW}|| > ||A_{HIGH}||$ we obtain $m_{ij} = ||A_{HIGH}||$.

Thus, the resulting matrix has the following form:

$$M = \begin{pmatrix} 1 & 1 & 1 & 1 & \cdots & 1 \\ 1 & 2 & 2 & 2 & \cdots & 2 \\ 2 & 2 & 3 & 3 & \cdots & 3 \\ 3 & 3 & 3 & 4 & \cdots & 4 \\ \vdots & \vdots & \vdots & \ddots & \ddots & \vdots \\ 2^{\frac{n}{2}}-2 & 2^{\frac{n}{2}}-2 & \cdots & \cdots & 2^{\frac{n}{2}}-2 & 2^{\frac{n}{2}}-1 \end{pmatrix}.$$

To prove that this matrix M has full rank we apply column transformations starting with a subtraction of the first column from all other columns. For the resulting matrix the submatrix consisting of the last $2^{\frac{n}{2}} - 2$ columns and the last $2^{\frac{n}{2}} - 2$ rows is an upper triangular matrix with 1's on the diagonal. By additional column tranformations it follows directly that the matrix has maximum rank $2^{\frac{n}{2}} - 1$. Consequently, the rank of the original communication matrix and $dim(< V^f_{n+1} >)$ is in $\Omega(2^{\frac{n}{2}})$. We summarize:

Lemma 2 *A* WLCD *with interleaved variable ordering representing function DIV has at least size* $(2^{\frac{n}{2}} - 1)$.

4.2 The General Case

In the rest of this section we give an idea how the above proof can be extended to WLCDs with arbitrary variable order. Crucial for the proof in the case of the interleaved variable ordering was the fact that the outcome of DIV was depending on the assignments for corresponding variables in the upper and lower half of the a-variables. More precisely: For the interleaved variable ordering there are $n/2$ pairs of variables (a_i, a_j) with

- a_i (a_j) is an upper (a lower) a-variable
- a_i and a_j are split between the first and the second half of the variable order and
- the difference of the indices i and j is a constant offset $\frac{n}{2}$.

For arbitrary variable orders this is not always the case for the offset $\frac{n}{2}$, but if one modifies the offset to a number $\frac{n}{2} - p_0$, one can always find "enough" such pairs being separated by the considered variable ordering. A precise formulation of this (purely combinatorial) property, its proof (and the application to lower bounds for multiplication) has already been given by Bryant in [4]. Using this property the proof for the interleaved variable ordering can be modified by specification of a "similar" communication submatrix. Consideration of the rank of this submatrix then leads to the following result (for details of the proof see [16]):

Theorem 3 *A* WLCD *for function DIV has at least size $2^{\frac{n}{16}} - 1$ (regardless of the variable order).*

Using Theorems 3 and 1 we finally obtain:

Corollary 1 MTBDD*s,* EVBDD*s,* BMD*s,* *BMD*s,* HDD*s,* K*BMD*s and* *PHDD*s require representations of size $\Omega(2^{\frac{n}{16}})$ for division (regardless of the variable order).*

5 Conclusions

We proved an exponential lower bound on the size of word-level representations for integer dividers. The proof could be done "simultaneously" for all word-level DDs by the introduction of Word-Level Linear Combination Diagrams (WLCDs) as a generic word-level DD. They turned out to be a powerful tool to characterize the limits of the word-level DD-concept.

Concerning division our result gives the following hints for future work: Since word-level DDs are not suitable as a data structure at least as long as they are used for the representation of the input-output behaviour, new methods have to be developed. If existing DDs are still to be used, e.g. the structure of the circuit might be considered to check whether a hierachical substitution based approach is feasible. On the other hand, it is an interesting open question, which type of (DD-similar) data structure is powerful enough to allow polynomial representation of division and efficient manipulation for verification at the same time.

References

[1] R.I. Bahar, E.A. Frohm, C.M. Gaona, G.D. Hachtel, E. Macii, A. Pardo, and F. Somenzi. Algebraic decision diagrams and their application. In *Int'l Conf. on CAD*, pages 188–191, 1993.

[2] B. Becker, R. Drechsler, and R. Enders. On the computational power of bit-level and word-level decision diagrams. In *ASP Design Automation Conf.*, pages 461–467, 1997.

[3] R.E. Bryant. Graph - based algorithms for Boolean function manipulation. *IEEE Trans. on Comp.*, 35(8):677–691, 1986.

[4] R.E. Bryant. On the complexity of VLSI implementations and graph representations of Boolean functions with application to integer multiplication. *IEEE Trans. on Comp.*, 40:205–213, 1991.

[5] R.E. Bryant. Binary decision diagrams and beyond: Enabeling techniques for formal verification. In *Int'l Conf. on CAD*, pages 236–243, 1995.

[6] R.E. Bryant and Y.-A. Chen. Verification of arithmetic functions with binary moment diagrams. In *Design Automation Conf.*, pages 535–541, 1995.

[7] Y.-A. Chen and R.E. Bryant. *PHDD: An efficient graph representation for floating point circuit verification. In *Int'l Conf. on CAD*, pages 2–7, 1997.

[8] E.M. Clarke, M. Fujita, and X. Zhao. Hybrid decision diagrams - overcoming the limitations of MTBDDs and BMDs. In *Int'l Conf. on CAD*, pages 159–163, 1995.

[9] E.M. Clarke, K.L. McMillan, X. Zhao, M. Fujita, and J. Yang. Spectral transforms for large Boolean functions with application to technology mapping. In *Design Automation Conf.*, pages 54–60, 1993.

[10] R. Drechsler, B. Becker, and S. Ruppertz. K*BMDs: A new data structure for verification. In *European Design & Test Conf.*, pages 2–8, 1996.

[11] R. Drechsler, A. Sarabi, M. Theobald, B. Becker, and M.A. Perkowski. Efficient representation and manipulation of switching functions based on ordered Kronecker functional decision diagrams. In *Design Automation Conf.*, pages 415–419, 1994.

[12] U. Kebschull, E. Schubert, and W. Rosenstiel. Multilevel logic synthesis based on functional decision diagrams. In *European Conf. on Design Automation*, pages 43–47, 1992.

[13] Y.-T. Lai and S. Sastry. Edge-valued binary decision diagrams for multi-level hierarchical verification. In *Design Automation Conf.*, pages 608–613, 1992.

[14] S. Malik, A.R. Wang, R.K. Brayton, and A.L. Sangiovanni-Vincentelli. Logic verification using binary decision diagrams in a logic synthesis environment. In *Int'l Conf. on CAD*, pages 6–9, 1988.

[15] M. Nakanishi. An exponential lower bound on the size of a binary moment diagram representing division. Master's thesis, Osaka University, 2 1998.

[16] C. Scholl, B. Becker, and T.M. Weis. Word-level decision diagrams, WLCDs and division. Technical Report 102, Albert-Ludwigs-University, Freiburg, May 1998.

[17] S. Waack. *On the Descriptive and Algorithmic Power of Parity Ordered Binary Decision Diagrams*, volume 1200 of *LNCS*. Symp. on Theoretical Aspects of Comp. Science, 1997.