On the Representational Power of Bit-Level and Word-Level Decision Diagrams

Bernd Becker Rolf Drechsler

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

Reinhard Enders

Siemens AG
Corporate Research and Development
81730 München, Germany
email: Reinhard.Enders@zfe.siemens.de

Abstract

Abstract— Several types of Decision Diagrams (DDs) have have been proposed in the area of Computer Aided Design (CAD), among them being bit-level DDs like OBDDs, OFDDs and OKFDDs. While the aforementioned types of DDs are suitable for representing Boolean functions at the bit-level and have proved useful for a lot of applications in CAD, recently DDs to represent integer-valued functions, like MTBDDs (=ADDs), EVBDDs, FEVBDDs, (*)BMDs, HDDs (=KBMDs), and K*BMDs, attract more and more interest, e.g., using *BMDs it was for the first time possible to verify multipliers of bit length up to $n = 256$.

In this paper we clarify the representational power of these DD classes. Several (inclusion) relations and (exponential) gaps between specific classes differing in the availability of additive and/or multiplicative edge weights and in the choice of decomposition types are shown. It turns out for example, that K(*)BMDs, a generalization of OKFDDs to the word-level, also "include" OBDDs, MTBDDs and (*)BMDs. On the other hand, it is demonstrated that a restriction of the K(*)BMD concept to subclasses, such as OBDDs, MTBDDs, (*)BMDs as well, results in families of functions which lose their efficient representation.

I. Introduction

One of the most important tasks during the construction and design of *Integrated Circuits* (ICs) is the proof of correctness, i.e. the check whether the design fulfills its specification.

In the last few years several methods based on *Decision Diagrams* (DDs) have been proposed [20, 18, 7] and in the meantime are used in industrial applications [14, 10, 1]. The most popular data structure in this context are *Ordered Binary Decision Diagrams* (OBDDs) [5].

The major drawback of OBDDs is that they often cannot represent the functions efficiently; this also holds for functions with high practical relevance, like multipliers [6].

Consequently, several variations of OBDDs have been proposed. Among these variations are those that use recursive application of three different decomposition types, the Shannon decomposition and the (positive and negative) Davio decomposition. This leads to the definition of *Ordered Kronecker Functional Decision Diagrams* (OKFDDs) [16], that, based on a formalization of the notion "decomposition type", have been proven to be the most general ordered DD for Boolean functions [3] combining the advantages of OBDDs that only use the Shannon decomposition and of *Ordered Functional Decision Diagrams* (OFDDs) that only use Davio decompositions. Furthermore, it was shown in [4] that OKFDDs in full generality are necessary to obtain succinct representations.

While the DDs mentioned above are suitable for representing Boolean functions at the bit-level and have proved useful for a lot of applications in CAD, recently, DDs to represent integer-valued functions attract more and more interest. In particular, *Binary Moment Diagrams* (BMDs) and *Multiplicative BMDs* (*BMDs) [8, 9], that are based on an integer-valued positive Davio decomposition and operate on the word-level, have become important. Using *BMDs it was for the first time possible to verify multipliers of bit length up to $n = 256$. Unfortunately, *BMDs fail for the representation of Boolean functions that can easily be represented using OBDDs [17]. Thus, extensions of *BMDs are of interest. A first step in this direction has been proposed in [11, 13]. There BMDs are combined with *Multi-Terminal BDDs* (MTBDDs) [12]. The resulting DDs, called *Hybrid Decision Diagrams* (HDDs) in [11], are based on integer-valued generalizations of Shannon and Davio decompositions. They combine the advantages of MTBDDs and BMDs in a similar way as OKFDDs combine OBDDs and OFDDs. We therefore prefer to call this class of DDs *Kronecker BMDs* (KBMDs). With KBMDs it is e.g. possible to handle functions related to data bits as BMDs and those related to control signals as MTBDDs.

*Kronecker *BMDs* (K*BMDs) [15] also use integer-valued generalizations of Shannon and Davio decompositions and in addition allow edge values as it is possible with *BMDs, *Edge-Valued Binary Decision Diagrams* (EVBDDs) [19], and *Factored EBDDs* (FEVBDDs) [21]. We obtain a data structure, that is a generalization of *BMDs, OBDDs and FEVBDDs as well. Thus, on the one hand it is possible to represent functions efficiently, that have a good word-level description, on the the other hand K*BMDs are also applicable to verification problems at the bit-level.

In this paper we clarify the representational power of classes of DDs. We prove several exponential gaps between specific classes. It follows that with respect to efficient representation it is advantageous to use (additive and multiplicative) edge weights. Furthermore, data structures that only allow the Shannon decomposition or only allow the Davio decomposition are not sufficient. DDs that allow different decomposition types in the same graph should be prefered, like OKFDDs, KBMDs and K*BMDs. Combining our results it follows that a re-

ASP-DAC '97
0-89791-851-7$5.00

striction of the K*BMD concept to subclasses, such as *BMDs and EVBDDs as well, results in families of functions which lose their efficient representation.

The paper is structured as follows: In Section II. we briefly review the definitions of the several types of DDs considered in this paper. In Section III. we then study the relation between these data structures. We finish with a resume of the results.

II. Decision Diagrams

In this section we introduce several classes of DDs. All these data structures 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. Furthermore, the underlying graph is *ordered*, i.e. the variables occur in the same order on all paths of the DD.

We first briefly review the basic notations and definitions of bit-level DDs like OKFDDs, OFDDs, OBDDs. Then word-level DDs like MTBDDs (=ADDs), BMDs, KBMDs, EVBDDs, FEVBDDs, *BMDs, and K*BMDs are introduced.

A. Bit-Level Decision Diagrams

We use OKFDDs as introduced in [16].

An OKFDD is a graph-based representation of a Boolean function $f : \mathbf{B}^n \to \mathbf{B}$, where at each non terminal node v (labeled with variable x) one of the following three decompositions is carried out:

$$\begin{aligned} f &= \overline{x} f_{low(v)} \oplus x f_{high(v)} && \textit{Shannon } (S) \\ f &= f_{low(v)} \oplus x f_{high(v)} && \textit{positive Davio } (pD) \\ f &= f_{low(v)} \oplus \overline{x} f_{high(v)} && \textit{negative Davio } (nD) \end{aligned}$$

(f is the function represented at node v, $f_{low(v)}$ ($f_{high(v)}$) denotes the function represented by the *low*-edge (*high*-edge) of v. $\oplus$ is the Boolean Exclusive OR operation.) The recursion stops at terminal nodes labeled with 0 or 1.

Decomposition types are associated to the n Boolean variables $x_1, x_2, \ldots, x_n$ with the help of a *Decomposition Type List* (DTL) $d := (d_1, \ldots, d_n)$ where $d_i \in \{S, pD, nD\}$, i.e. d_i provides the decomposition type for variable x_i ($i = 1, \ldots, n$). An OKFDD with DTL $(S, \ldots, S)$, i.e. all nodes are Shannon nodes, is an OBDD. An OKFDD with DTL $d := (d_1, \ldots, d_n)$ where $d_i \in \{pD, nD\}$, i.e. all nodes are Davio nodes, is an OFDD. Analogously, OpFDDs (OnFDDs) are OKFDDs with only positive (negative) Davio nodes.

On OKFDDs reductions can be defined resulting in canonical representations for not only OBDDs and OFDDs, but also for OKFDDs. The size of an OKFDD can be further reduced, if *Complement Edges* (CEs), that are well known from OBDDs, are used. Then a node is used to represent a function and its complement at the same time.

The scenario of bit-level DDs considered in this paper is given in the upper part of Fig. 1. A line indicates a subset relation, that is directly given by the definitions, e.g., OBDDs are a subset of OKFDDs.

B. Word-Level Decision Diagrams

We now briefly introduce Decision Diagrams for the representation of integer-valued functions $f : \mathbf{B}^n \to \mathbf{Z}$. (For more details see [12, 2, 8, 9, 11, 19, 15].)

All word-level DDs considered here are also graph-based representations where the underlying graph is ordered. As for bit-level DDs, reductions can be defined resulting in canonical representations for all types of DDs considered here.

We start with word-level DDs making no use of edge values.

B.1 Word-Level DDs without edge values

MTBDDs (=ADDs) are based on the (integer-valued) *Shannon decomposition* (S) and allow terminal nodes labeled with integer values. BMDs make use of the (integer-valued) *positive Davio decomposition* (pD) and allow terminal nodes labeled with integer values, i.e., they are the integer-valued generalization of OpFDDs.

KBMDs (or HDDs as they are called by Clarke et al. [11]) try to combine the advantages of MTBDDs and BMDs. Analogously to OKFDDs at the bit-level, different decomposition types per variable can be used. Since we consider integer-valued functions a lot of differing decomposition types are possible. They can be defined by the set $\mathbf{Z}_{2,2}$ of non singular 2×2 matrices over $\mathbf{Z}$ [11]. As for OKFDDs decomposition types are associated to the n Boolean variables with the help of a *Decomposition Type List* (DTL) $d := (d_1, \ldots, d_n)$ where $d_i \in \mathbf{Z}_{2,2}$, i.e. for each variable one fixed decomposition is chosen. Following [11] the matrices corresponding to Shannon (S), positive Davio (pD) and negative Davio (nD), respectively, are

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

The middle part of Fig. 1 depicts the scenario of word-level DDs without edge values.

B.2 Word-Level DDs with edge values

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.

EVBDDs are MTBDDs where a constant a is *added* to the function being represented. Thus, in the EVBDD an edge with weight a to a node v labeled with variable x represents the function $< a, f >= a + (1 - x) f_{low(v)} + x f_{high(v)}$. (As before f is the function represented at node v, $f_{low(v)}$ ($f_{high(v)}$) denotes the function represented by the *low*-edge (*high*-edge) of v. $-$ and $+$ denote the usual operations in the ring $\mathbf{Z}$.)

*BMDs are a generalization of BMDs in the sense that they allow multiplicative edge weights: the values at the edges are multiplied with the functions represented. Thus, an edge with weight m to a node v in a *BMD represents the function $< m, f >= m(f_{low(v)} + x f_{high(v)})$.

K*BMDs differ from KBMDs in the fact that they allow the use of integer weights, additive **and** multiplicative weights in parallel. An anlogous concept is realized for MTBDDs in [21]. There EVBDDs are supplied with an additional multiplicative weight resulting in *Factored EVBDDs* (FEVBDDs). K*BMDs (and FEVBDDs) make use of the following type of representation:

$$< (a, m), f >:= a + mf$$

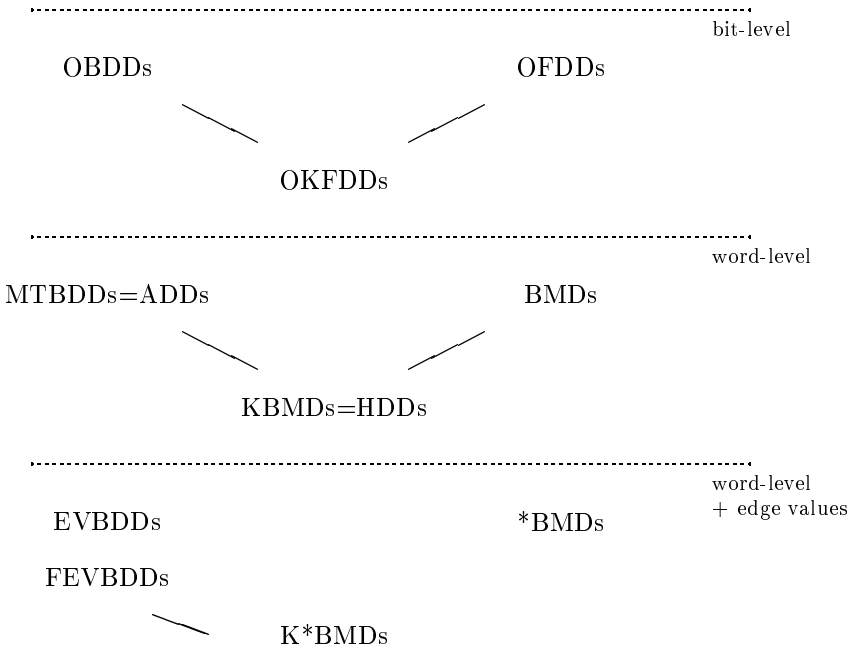

Fig. 1. World of Ordered Decision Diagrams

In contrast to FEVBDDs, which are based on Shannon decomposition, K*BMDs allow differing decomposition types per variable. In the case of Shannon decomposition, positive and negative Davio decomposition the function represented at a node v is then given by

$$\begin{aligned} < (a,m), f > &= a + m((1-x)f_{low(v)} + xf_{high(v)}) \\ < (a,m), f > &= a + m(f_{low(v)} + xf_{high(v)}) \\ < (a,m), f > &= a + m(f_{low(v)} + (1-x)f_{high(v)}) \end{aligned}$$

To make DDs with edge values a canonical representation some further restrictions on the graph with respect to weights are required. For simplicity, here we only comment on these restrictions for the case of K*BMDs. (For other DD types the restrictions are similar.) Basically the following is required: There exists only one leaf and this leaf is labeled 0, the *low*-edge of a node has additive weight 0 and the remaining weights have *greatest common divisor* 1.

The lower part of Fig. 1 gives the edge-valued word-level DDs considered here.

III. Relations between Decision Diagrams

In this section we study the representational power of these DDs in more detail. In particular, we are interested in exponential trade-offs of the following kind: "For the representation of bit-level functions OFDDs are at least as small as *BMDs. There exist functions where the best *BMD representation is exponentially larger than the corresponding EVBDD representation."

A. Basic Properties

We start with some simple observations.

It follows directly from the definition that MTBDDs for Boolean functions are isomorphic to OBDDs (without CEs).

According to the normalization rules for K*BMDs (and FEVBDDs) it follows easily that an edge weight $(1,-1)$ at the *high*-edge of a node that represents a Boolean function corresponds to a complement edge. Thus, in K*BMDs for Boolean functions consisting of only Shannon nodes (= FEVBDDs) all inner nodes also represent only Boolean functions (in contrast to *BMDs). It follows, that K*BMDs (for the representation of Boolean functions) are isomorphic to OBDDs with CEs, if the Shannon decomposition is carried out in each node.

If K(*)BMDs with decomposition types S, pD, nD are used for the representation of Boolean functions, they are always larger (or equal) in size than OKFDDs (with CEs), since OKFDDs result from KBMDs and K*BMDs by performing a modulo 2 operation on the terminals and nodes, respectively, and performing a reduce operation. As a special case it follows, that *BMDs (EVBDDs) for Boolean functions are always larger (or equal) in size than OpFDDs (OBDDs).

It follows from the normalization rule that K*BMDs have only one node for the representation of a single variable. In EVBDDs and *BMDs several nodes might be used. In general K*BMDs need a smaller number of nodes than *BMDs for the representation of functions, since they have a larger expressive power (see the following sections), but the space needed for each node is slightly larger due to the additive and multiplicative weights. In the following subsection we will show that merely the use of edge values leads to smaller (sometimes exponentially smaller) representation sizes (counted in the number of nodes).

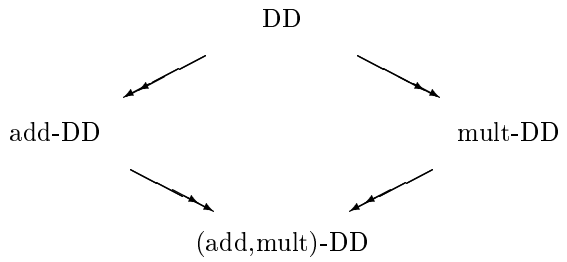

Fig. 2. DDs with differing edge values.

B. Why to Use Edge Values

For the following let DD be any of the word-level DDs without edge values as introduced in the previous section. We now consider three classes resulting from DD by the differing types of edge values:

add-DD: DD is augmented by additive weights. (E.g., MTBDDs and EVBDDs are related in this way.)

mult-DD: The DD is augmented by multiplicative weights. (For an example consider BMDs and *BMDs.)

(add,mult)-DD: DD is augmented by additive and multiplicative weights. (K*BMDs result from KBMDs by doing this.)

It can be shown inductively that common subfunctions in DD remain common subfunctions after the augmentation of the DD with edge values. It may happen that a "lot" of new common subfunctions result when edge values are allowed.

As an example consider the function

$$m(x_1, x_2, \ldots, x_n) := \prod_{i=1}^{n} p_i^{x_i} + 1$$

where p_i is the i^{th} prime number. It is easy to see that $m(x)$ has linear (add,mult)-MTBDD size (= FEVBDD size) but only exponential MTBDDs, add-MTBDDs (=EVBDDs) and mult-MTBDDs (= MTBDDs with multiplicative weights).

In a similar way one can show exponential gaps for all cases, where edge values are added. For illustration see Fig. 2, where a double arrow from A to B means that, given an ordering of the variables and a DTL, the representation for any function f in class A is *always larger (or equal), sometimes exponentially larger* than in class B.

We summarize the results obtained in Subsections III.A and III.B. (For illustration see Fig. 3. An arrow from A to B indicates that representation in A is always larger or equal as in B. A double arrow has the same meaning as in Fig. 2. Solid lines are used to denote the inclusion (of OBDDs in FEVBDDs and K*BMDs).) Until now we have shown some "vertical relations":

- Going from bit-level DDs to word-level DDs in general increases the size for the representation of Boolean functions.
- This increase can be "softened" by the introduction of edge values which reduce reduction size sometimes exponentially (compared to the DD version without weights).
- On the other hand, the gaps may remain exponential even in the presence of weights. See the double arrows between K(*)BMDs and OKFDDs ((*)BMDs and OFDDs). (This will be shown in Theorem 1 given in the next section.)
- FEVBDDs and K*BMDs are the only word-level DDs that *contain* a bit-level DD type (OBDDs) and thus *guarantee* at least the same representation size as OBDDs.

C. Why to Use different Decomposition Types

In this subsection we discuss gaps resulting from the use of different decomposition types. This will help us to clarify the "horizontal" relations between DDs.

The representation size for a function may heavily depend on the decomposition type. This can be demonstrated by the so-called *clique functions* which are shortly introduced in the following:

Consider n nodes $1, 2, \ldots, n$ and $n(n-1)/2$ Boolean variables $x_{i,j}$ $(1 \leq i < j \leq n)$. Then an assignment $x = (x_{1,2}, x_{1,3}, \ldots, x_{n-1,n}) \in \mathbf{B}^{n(n-1)/2}$ defines an undirected graph $G_x = (V, E_x)$ with $V = \{1, \ldots, n\}$ and $E_x = \{(i,j) | x_{i,j} = 1\}$. A group consisting of 3 nodes is called a 3-clique (or clique for simplicity), iff every two of them are connected by an edge. Let $d = (d_{1,2}, d_{1,3}, \ldots, d_{n-1,n})$ be a DTL. Then the *polarity function* $pol_d : \mathbf{B}^{n(n-1)/2} \to \mathbf{B}^{n(n-1)/2}$ is defined by $pol_d(x_{1,2}, \ldots, x_{n-1,n}) := (x^d_{1,2}, \ldots, x^d_{n-1,n})$ with

$$x^d_{i,j} := \begin{cases} x_{i,j} & : \quad d_{i,j} = S, d_{i,j} = pD \\ \overline{x}_{i,j} & : \quad d_{i,j} = nD \end{cases}$$

For a fixed DTL d we consider the *1-clique-function* 1-$cl^d_{n,3}$, the *parity-clique-function* $\oplus$-$cl^d_{n,3}$ and the *number-clique-function* #-$cl^d_{n,3}$ defined by

- 1-$cl^d_{n,3}(x) = 1$, iff $G_{pol_d(x)}$ contains exactly three edges and these edges build a 3-clique.
- $\oplus$-$cl^d_{n,3}(x) = 1$, iff $G_{pol_d(x)}$ contains an odd number of 3-cliques.
- #-$cl^d_{n,3}(x) = k$, iff $G_{pol_d(x)}$ contains exactly k 3-cliques.

In Table I we provide some experimental data giving OBDD (OFDD, EVBDD, *BMD, K*BMD) representation sizes of 1-$cl^d_{n,3}$, $\oplus$-$cl^d_{n,3}$ and #-$cl^d_{n,3}$ ($d = (pD, pD, \ldots, pD)$) for varying n. An 'X' symbolizes that the function has an integer range and the data structure is Boolean only. A '-' symbolizes that the construction exceeded a node limit of 100.000 nodes. Only the "pure" DDs are considered for K*BMDs, i.e. mixing of decomposition types is not yet considered. For all functions we used the initial variable ordering. Our experiments show, that Shannon-based DDs can handle 1-$cl^d_{n,3}$. #-$cl^d_{n,3}$ is

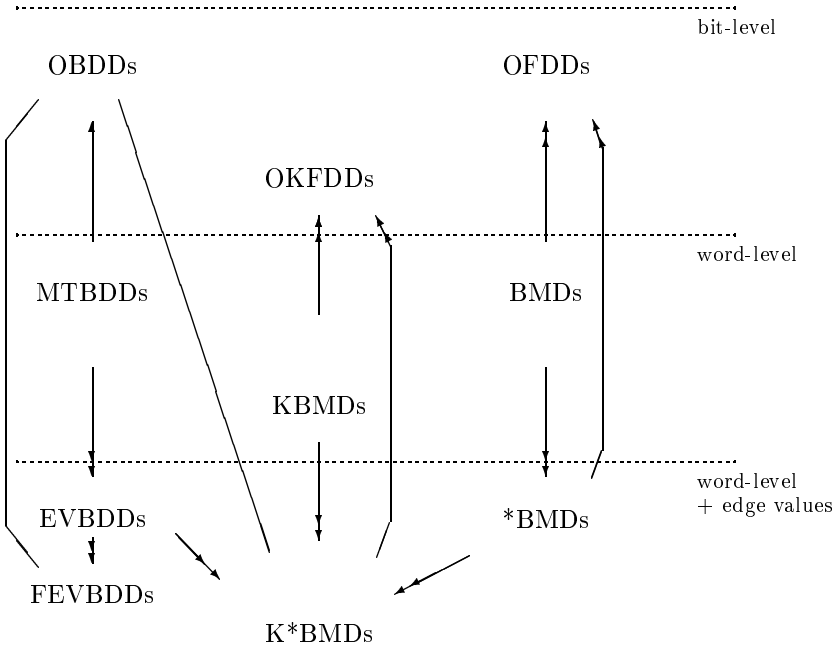

Fig. 3. "Vertical" relations between different types of DDs

easily represented by Davio-based word-level representations, while Shannon-based representations fail already for $n = 8$. $\oplus$-$cl^d_{n,3}$ can only be represented efficiently using OFDDs (= bit-level Davio-based representations). Word-level representations (even the word-level Davio-based ones) fail.

The (more) theoretical counterpart of these experiments is provided by a number of exponential gaps between specific classes of DDs given in the following. For bit-level DDs these gaps have been already shown in [4], they are briefly repeated here for the sake of completeness. Furthermore, we demonstrate how the results of [17] and [4] can be extended to general word-level DDs.

The key to the proofs of all theorems given in the following is an investigation of the relation between functions which are represented by the same (bit-level or word-level) DD G. Of course, changing the DTL d for G in general also changes the function represented by the DD. Nevertheless, it can be shown that the functions are related by a bijective operator, called the generalized τ-operator. Because of its importance we shortly introduce this operator. All further proofs in this paper are left out due to page limitation.

Defining the *generalized τ-operator* τ_d for a function f and DTL d by

$$\tau_d(f)(x) := \bigoplus_{y \leq_d x} f(y)$$

it can be shown that the the functions associated to DDs by different interpretations can be computed from each other through the generalized τ-operator and its inverse.[1]

Using this it is possible to transfer results between different types of DDs. In particular, the existence of families of functions that provide exponential gaps can be proven.

Theorem 1 (Gaps: Shannon versus Davio)
For any order π of the variables $x_{i,j}$ and any DTL $d = (d_{1,2}, \ldots, d_{n-1,n}) \in \{pD, nD\}^{n(n-1)/2}$ it holds:

i.) 1-$cl^d_{n,3}$ has OBDDs, MTBDDs (=ADDs), EVBDDs, FEVBDDs of polynomial size ($O(n^5)$), but only OFDDs, KBMDs, K*BMDs (with DTL d) of exponential size ($2^{\Omega(n^2)}$).

ii.) $\oplus$-$cl^d_{n,3}$ has OFDDs (with DTL d) of polynomial size $O(n^3)$, but only OBDDs, *BMDs, K(*)BMDs with DTL d of exponential size ($2^{\Omega(n)}$).

iii.) #-$cl^d_{n,3}$ has KBMDs, K*BMDs (with DTL d) of polynomial size ($O(n^3)$), but only MTBDDs (=ADDs), EVBDDs, FEVBDDs of exponential size ($2^{\Omega(n^2)}$).

We illustrate relations obtained by this theorem in Fig. 4). (The arrows denote potential gaps.) It follows that depending on the function it is advantageous to consider both, Shannon-based DDs and Davio-based DDs. Thus, DDs combining Shannon and Davio decompositions are helpful.

D. Why to use "many" DTLs

As demonstrated in the previous subsection OKFDDs, KBMDs, K*BMDs with S-nodes and OKFDDs, KBMDs,

[1] $\bigoplus$ (in the formula) denotes the sum in the Boolean or integer domain depending on whether we consider bit- or word-level DDs. $y \leq_d x$ ("smaller with respect to d") iff all components i satisfy $y_i \leq_{d_i} x_i$, where $y_i \leq_{d_i} x_i$ means $y_i \leq x_i$ ($y_i \leq \overline{x_i}$, $y_i = x_i$) iff $d_i = pD$ ($d_i = nD$, $d_i = S$).

TABLE I
COMPARISON BETWEEN DIFFERENT DD TYPES

name	n	inp	OBDD	OFDD	*BMD	K*BMD		
						S	pD	nD
1-$cl^d_{n,3}$	5	10	47	114	171	47	154	65
	7	21	207	6248	19676	207	19399	275
	8	28	367	-	-	367	-	477
	10	45	947	-	-	947	-	1185
$\oplus$-$cl^d_{n,3}$	5	10	94	22	123	94	112	217
	7	21	4005	65	25410	4005	24647	-
	8	28	36392	98	-	36392	-	-
	10	45	-	192	-	-	-	-
#-$cl^d_{n,3}$	5	10	X	X	22	108	22	33
	7	21	X	X	65	6452	65	94
	8	28	X	X	98	-	98	139
	10	45	X	X	192	-	192	263

K*BMDs with D-nodes should be considered. We now show that it also makes sense to mix differing decomposition types within one and the same DD. Furthermore, it is shown that the restriction to a few DTLs reduces the power of DDs.

We first provide an exponential gap for DDs with DTL $d \in \{pD, nD\}^n$ and DDs with DTL $\overline{d}$. Here, the DTL $\overline{d}$ results from the DTL d by changing the decomposition types from pD to nD and vice versa.

Theorem 2 (Gaps: Davio versus inverse Davio) Consider any order π of the variables $x_{i,j}$ and any DTL $d \in \{pD, nD\}^{n(n-1)/2}$. Define the DTL $\overline{d}$ by $\overline{d}_i := pD$ (nD) if $d_i = nD$ (pD). Then it holds:

> 1-$cl^d_{n,3}$ has K(*)BMDs with DTL $\overline{d}$ of polynomial size, but only K(*)BMDs with DTL d of exponential size.

The last two theorems are the basis for an important corollary.

Corollary 1 There exists an exponential gap between two DTLs, d_1 and d_2, differing in all components.

Summing up the results so far, we have shown that it is advantageous to consider K(*)BMDs with several different DTLs. A restriction to only one DTL results in functions that lose their efficient representation. We now consider the problem, whether a restriction to a few DTLs reduces the power of K(*)BMDs. We show that as in the bit-level case a few DTLs are not as powerful as general K(*)BMDs. This also implies that mixing of decomposition types in one DTL is important.

Theorem 3 (Gaps for restricted DTLs) Consider a set $D = \{d^1, d^2, \ldots, d^k\}$ ($k \in \mathbf{N}$ constant) of DTLs $d^j = (d^j_1, d^j_2, \ldots, d^j_n)$ with $d^j_i \in \{S, pD, nD\}$. Then the following holds:

> There exists a function f_n over n variables, such that each K*BMD (with DTL $d^j, 1 \leq j \leq k$) for f_n has exponential size, while there exist K*BMDs for f_n of polynomial size.

At the end of this section we want to mention some further points of interest not discussed in this paper. Besides representation size also the complexity of synthesis operations on DDs is of interest. The exponential gaps proved in this paper can be used to determine the worst case complexity of operations. It can for example be shown that already for "simple" word-level DDs, like BMDs, *BMDs, FEVBDDs, multiplication has exponential worst case complexity. Thus, conciseness of representation (at least in some cases) has to be paid for with a decrease of efficiency in the operations.

IV. CONCLUSIONS

In this paper we presented some results to clarify the relation between different types of Decision Diagrams - data structures frequently used in CAD applications.

We proved several (exponential) gaps between specific classes. A good summary is provided by the combination of Fig. 1, 3, 4:

In general the representation of Boolean functions with word-level DDs leads to an increase in size. K(*)BMDs and FEVBDDs have the advantage to "link" word-level and bit-level representation in the sense, that OBDDs are contained as a subset in both.

In the field of word-level DDs (additive and multiplicative) edge weights should be used to reduce the size.

For bit-level and word-level representations as well, data structures that only allow Shannon decompositions (only allow Davio decompositions) are not sufficient. DDs that provide the possibility to use different decomposition types in the same graph should be prefered, like OKFDDs, KBMDs and K*BMDs. We also showed that a restriction of the K(*)BMD concept to a fixed number of subclasses, e.g. BMDs, MTBDDs, *BMDs and EVBDDs as well, results in families of functions which lose their efficient representation.

REFERENCES

[1] D.P. Appenzeller and A. Kuehlmann. Formal verification of a PowerPC microprocessor. In *Int'l Conf. on Comp. Design*, pages 79–84, 1995.

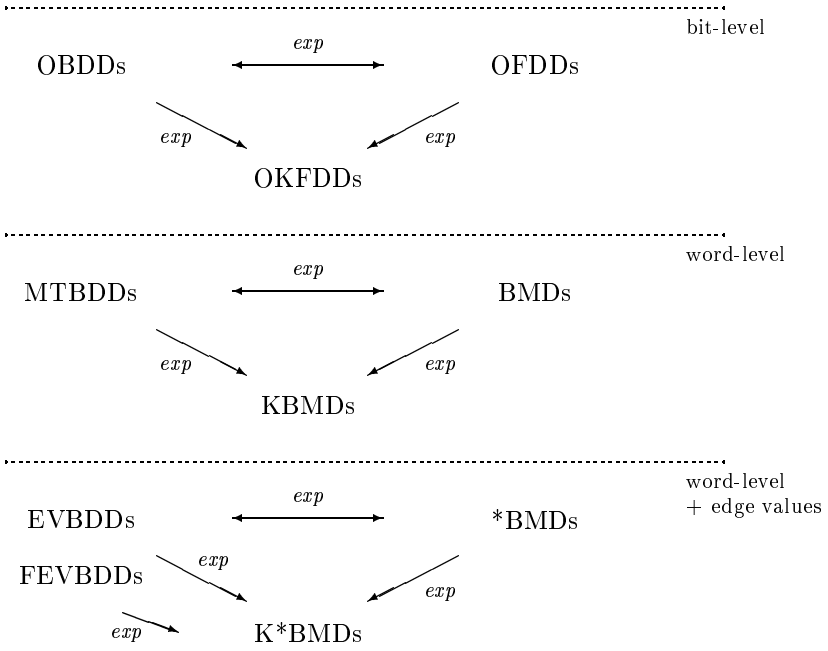

Fig. 4. Exponential "horizontal" gaps between different types of DDs.

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

[3] B. Becker and R. Drechsler. How many decomposition types do we need? In *European Design & Test Conf.*, pages 438–443, 1995.

[4] B. Becker, R. Drechsler, and M. Theobald. OKFDDs versus OBDDs and OFDDs. In *ICALP, LNCS 944*, pages 475–486, 1995.

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

[6] 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.

[7] R.E. Bryant. Symbolic boolean manipulation with ordered binary decision diagrams. *ACM, Comp. Surveys*, 24:293–318, 1992.

[8] R.E. Bryant and Y.-A. Chen. Verification of arithmetic functions with binary moment diagrams. Technical report, CMU-CS-94-160, 1994.

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

[10] J.R. Burch, E.M. Clarke, K.L. McMillan, D.L. Dill, and L.J. Hwang. Symbolic model checking: 10^{20} states and beyond. *Information and Computation*, 98(2):142–170, 1992.

[11] 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.

[12] E.M. Clarke, K. 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.

[13] E.M. Clarke and X. Zhao. Word level symbolic model checking - a new approach for verifying arithmetic circuits. Technical Report CMU-CS-95-161, 1995.

[14] O. Coudert, C. Berthet, and J.C. Madre. Verification of sequential machines based on symbolic execution. In *Automatic Verification Methods for Finite State Systems, LNCS 407*, pages 365–373, 1989.

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

[16] 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.

[17] R. Enders. Note on the complexity of binary moment diagram representations. *IFIP WG 10.5 Workshop on Applications of the Reed-Muller Expansion in Circuit Design*, pages 191–197, 1995.

[18] M. Fujita, H. Fujisawa, and N. Kawato. Evaluation and improvements of boolean comparison method based on binary decision diagrams. In *Int'l Conf. on CAD*, pages 2–5, 1988.

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

[20] 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.

[21] P. Tafertshofer and M. Pedram. Factored edge-valued binary decision diagrams. to appear in *Formal Methods in System Design*, 1996.