On the Evolution of the Worst-Case OBDD Size

Clemens Gröpl* Hans Jürgen Prömel** Anand Srivastav***

Abstract

We prove matching lower and upper bounds on the worst-case OBDD size of a Boolean function, revealing an interesting oscillating behavior.

Keywords: Data structure, ordered binary decision diagram, Boolean function.

1 Introduction

The ordered binary decision diagram (OBDD) is a representation of Boolean functions $f\colon \{0,1\}^n \to \{0,1\}$ that combines compactness with algorithmic manageability. The history of OBDDs dates back to Lee's seminal article from 1959 [10], but their algorithmic properties were recognized only in the early eighties by Bryant [3, 4]. Nowadays, several efficient implementations are available, e. g. [2, 13] and OBDDs are in widespread use.

An OBDD is a branching program in which the variables are tested only once and according to some fixed variable ordering. More explicitly, an OBDD is an acyclic directed graph with one root and two terminals 0 and 1, in which each internal (non-terminal) node has two outgoing edges pointing to successor nodes and is labeled with a variable. To evaluate the function represented by an OBDD, computation starts at the root node and in each step follows one of the two outgoing edges according to the value of the variable of the node being visited, until finally a terminal is reached. A *level* is the set of all nodes which are labeled with a particular variable. The variable ordering implies that an OBDDs consists of levels stacked on top of each other. An OBDD is *reduced,* if it contains no nodes representing the same subfunction. OBDDs are assumed to be reduced unless stated otherwise. The OBDD for a Boolean function is unique up to the choice of the variable ordering. We can obtain the ordered binary decision diagram of a Boolean function from its ordered binary decision tree by applying two reduction rules. The *merging rule* allows us to identify nodes that test the same variable and have coinciding successor nodes. The *deletion rule* asserts that nodes with both outgoing edges pointing to the same successor can be removed, because the function they represent does not depend on the variable that is about to be tested. The *quasi-reduced* ordered binary decision diagram (qOBDD) results from applying (only) the merging rule to its binary decision tree. Like the OBDD, it is uniquely determined for each variable ordering, and from the qOBDD of a function we obtain the OBDD using the deletion rule.

Let us denote the worst-case size of the reduced OBDD of a Boolean function of n variables by $W'(n)$. Already Lee [10] proved that the worst-case OBDD size is at most $4 \cdot 2^n/n - 2$. He also showed that Boolean functions exist whose BDD size is at least $\frac{1}{2} \cdot 2^n/n + 1$. A BDD is defined like an OBDD, but the read-once

---

[*] Corresponding author. Humboldt-Universität zu Berlin, Institut für Informatik, 10099 Berlin, Germany. Email: groepl@informatik.hu-berlin.de. Most of this work was done while this author was with graduate school "Algorithmische Diskrete Mathematik", supported by Deutsche Forschungsgemeinschaft, grant GRK 219/2-97.

[**] Humboldt-Universität zu Berlin, Institut für Informatik, 10099 Berlin, Germany. Email: proemel@informatik.hu-berlin.de. Supported by Deutsche Forschungsgemeinschaft, grant Pr 296/3-2.

[***] Christian-Albrechts-Universität zu Kiel, Mathematisches Seminar II, Ludewig-Meyn-Straße 4, 24098 Kiel, Germany. Email: asr@numerik.uni-kiel.de. Most of this work was done while this author was supported by Deutsche Forschungsgemeinschaft, grant Pr 296/3-2.

property and the variable ordering requirement are dropped. Therefore, his lower bound also applies to OBDDs. The proof uses a variant of Shannon's classical counting argument [12] and shows that in fact almost all Boolean functions have such large BDDs.

Liaw and Lin [11] improved Lee's upper bound to $\bigl(2+o(1)\bigr)\cdot 2^n/n$. Their upper bound is attained for certain multiplexor functions with $n = 2^h + h$ variables. Minor errors were corrected by Heap and Mercer [9].

Breitbart, Hunt, and Rosenkrantz [1] showed that the worst-case BDD size is $\bigl(1+o(1)\bigr)\cdot 2^n/n$. Independently of [11] and [9], they also discovered the $\bigl(2+o(1)\bigr)\cdot 2^n/n$ upper bound for OBDDs and observed that the same construction yields a matching upper bound of $\bigl(1+o(1)\bigr)\cdot 2^n/n$ for certain values of n. They posed the problem "to more exactly characterize" W' [1, p. 57], and it seems that they (falsely) assumed that the upper bound $\bigl(1+o(1)\bigr)\cdot 2^n/n$ also holds for OBDDs on the analogy of their tight BDD result.

Notice that the results of [11] and [1] together imply that the ratio $W'(n)\big/(2^n/n)$ is bounded, but does not converge to a limit. Such an oscillating behavior does not occur for BDDs.

Heap and Mercer [9] gave an exact formula for $W'(n)$ (as a sum over the level sizes) and an explicitly defined series of functions that attains this value for each $n \in \mathbb{N}$. They also gave a diagram [9, Figure 1] of $W'(n)\big/(2^n/n)$ for $n \leqslant 160$ which reveals the oscillations, but did not comment on this fact. Moreover, although their diagram suggests that the right 'global' lower bound should be $\bigl(1+o(1)\bigr)\cdot 2^n/n$, the authors only proved that $W'(n) \geqslant \frac{1}{2}\cdot 2^n/n$.

In this article we give a precise analysis of the asymptotics and the evolution of the worst-case OBDD size for all n. We determine the asymptotic value of $W'(n)$ for all n, and the shape of the oscillations. It turns out to be advantageous not to compare $W'(n)$ with $2^n/n$, but with $2^{L(n)}$, where the function $L : \mathbb{R}_+ \to \mathbb{R}_+$ is defined by the equation $n = L(n) + \log L(n)$. In this way, we can remove the $o(1)$ terms in several places, thereby also improving the 'global' bounds from [11, 9, 1]. Finally, we discuss some consequences of our results and how they are related with recent results on the OBDD (and FBDD) size of *almost all* Boolean functions [15, 7, 5].

2 Preliminaries

2.1 Definitions

We consider ordered binary decision diagrams for a Boolean function f respecting the variable ordering $x_1, \dots, x_n$. Let us denote the quasireduced ordered binary decision diagram for f (where only the merging rule is available) by qOBDD(f). The nodes on level i of qOBDD(f) represent the different subfunctions of f which result from substituting the first $i-1$ variables $x_1, \dots, x_{i-1}$ by constants $c_1, \dots, c_{i-1}$. Clearly, the level sizes in qOBDDs are upper bounded by the growth rate of the decision tree, $k_i := 2^{i-1}$. In the lower part, another observation becomes important: The number of Boolean functions with $n-i+1$ variables is $m_i := 2^{2^{n-i+1}}$. See Figure 1. Similarly, we denote the (fully reduced) ordered binary decision diagram for f by OBDD(f). In OBDDs, a node is present only if the corresponding subfunction 'really' depends on the variable tested at that node. Hence we obtain a slightly smaller upper bound of $m'_i := m_i - m_{i+1}$ for the level width in the lower part of OBDDs. Let $w_i := \min\{k_i, m_i\}$ and $w'_i := \min\{k_i, m'_i\}$. The upper bounds w_i resp. w'_i are tight (see Lemma 1 below). So the worst-case size is $W(n) := \sum_{i=1}^{n} w_i$ for qOBDDs and $W'(n) := \sum_{i=1}^{n} w'_i$ for OBDDs. A rough estimate is $W(n), W'(n) = \Theta(2^n/n)$. A tight estimate cannot have this form, because the ratios $W(n)n/2^n$ and $W'(n)n/2^n$ oscillate in a periodic way.

2.2 Tight Bounds for the Level Sizes

The upper bounds w_i and w'_i are tight, as is shown in the following lemma proved in [9]. It is included here for completeness.

Lemma 1 *For all n, there exist Boolean functions f and f' of n variables such that for all $i \in [n]$, level i of qOBDD(f) (resp. OBDD(f)) has width w_i (resp. w'_i).*

Proof. We show how to construct reduced and quasireduced OBDDs matching the upper bounds, thus defining f and f'. Let i be the largest index such that $k_i \leqslant m_i$. For the top part of the OBDD, we take a decision tree with k_i terminals. Since $k_i \leqslant m_i$, we can assign different subfunctions depending on $n - i + 1$ variables to all the terminals of the decision tree. So we can guarantee that in the resulting OBDD, no mergings are possible *above* and *at* level i.

But how can we enforce that the upper bound is attained at the levels *below* i? We need to specify the way in which subfunctions are assigned to the leaf nodes of the top part tree in more detail. Think of a 'universal' shared OBDD U representing all Boolean functions of the lower variables $x_i, \ldots, x_n$. (A shared OBDD represents a family of Boolean functions, using the reduction rules in the obvious way.) In the reduced OBDD case, U has m_i nodes, including the sinks, whereas in the quasireduced OBDD case, the top level of U where x_i is tested (possibly redundantly) alone already has m_i nodes. In the reduced OBDD case, the terminals of the top part tree are replaced with nodes from the whole lower part, while in the quasireduced OBDD case, we only use the nodes at the top level of the lower part.

Since $2k_i = k_{i+1} > m_{i+1}$, we can choose the first $m_{i+1}/2$ subfunctions in such a way that the high- and low-successors of these nodes already 'cover' the second level of U (where x_{i+1} is tested) completely. The levels below $i + 1$ are 'full', because each node at level $j \geqslant i + 2$ has incoming edges from level $j - 1$. □

2.3 The Function L

Due to the fast growth and shrink rate of the upper bounds k_i and m_i on the level sizes, is clear that the most interesting part of a worst-case qOBDD is near a certain 'critical point' i_0 where the two bounds meet. Worst-case OBDDs look almost the same. We consider m_i instead of m'_i for technical reasons. The equation $k_i = m_i$ has no solution in closed form. We define the function L by the functional equation

$$L(n) + \log L(n) = n \tag{1}$$

and set

$$i_\delta := L(n) + \delta + 1\,. \tag{2}$$

Then we have

$$k_{i_\delta} = 2^{\delta + L} \quad \text{and} \quad m_{i_\delta} = 2^{2^{-\delta} L}\,. \tag{3}$$

With these definitions, i_0 is indeed the critical point, because

$$w_{i_0} = k_{i_0} = m_{i_0} = 2^{L(n)}\,.$$

Notice that i_0 is not an integer in general. It just marks the point where w_i (as a function of i) turns from growing exponentially to shrinking doubly exponentially. See Figure 1.

Earlier investigations dealt with $n - \log n$ in some way [15] or approximated $L(n)$ by other means [11]. In [9], $\lceil i_0 \rceil$ appeared implicitly in the form of $\min\{\, i \in \mathbb{N} \mid k_i \geqslant m_i \,\}$, but the authors did not provide an asymptotic for $\lceil i_0 \rceil - i_0$. We found it advantageous to deal directly with L, so that we can use the functional equation (1). It is easy to verify that $L(n) = n - \log n + o(1) \sim n$ and $2^{L(n)} \sim 2^n/n$. See also [5].

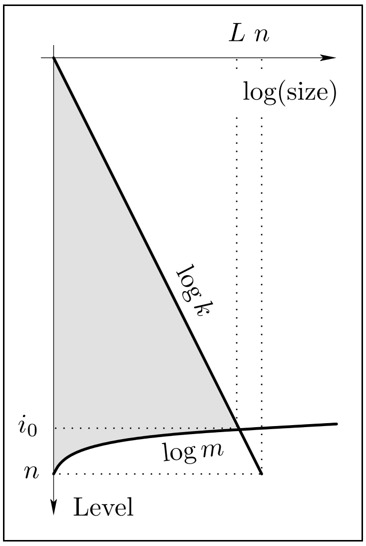

Figure 1: The worst-case shape of a qOBDD

3 Evolution of the Worst-Case Sizes

We start with an analysis of the qOBDD case. The results do not change by much when we include the effect of the deletion rule later.

3.1 Quasireduced OBDDs

By definition of the function L, we have

$$w_i = \begin{cases} k_i \,, & i \leqslant i_0 \,; \\ m_i \,, & i \geqslant i_0 \,. \end{cases} \tag{4}$$

To estimate W, we write out the sum and substitute (4).

$$\begin{aligned} W(n) &= \sum_{i=1}^{\lceil i_0 \rceil - 1} 2^{i-1} + \sum_{i=\lceil i_0 \rceil}^{n} 2^{2^{n-i+1}} \\ &= \sum_{i=0}^{\lceil L \rceil - 1} 2^{i} + \sum_{i=1}^{n-\lceil L \rceil} 2^{2^{i}} \\ &= 2^{\lceil L \rceil} + 2^{2^{n-\lceil L \rceil}} + \left| O\Big(2^{2^{n-\lceil L \rceil - 1}}\Big) \right| . \end{aligned} \tag{5}$$

Both leading terms are roughly of size 2^L. The exact value of $\lceil L \rceil$ is given by the next lemma (cf. [9, Lemma 3]).

Lemma 2

$$\big\lceil L\big(2^h + h + a\big) \big\rceil = \begin{cases} 2^h + a + 1 \,, & a \in \big[-2^{h-1} - 1 \,..\, -1\big] \;; \\ 2^h + a \,, & a \in \big[0 \,..\, 2^h\big] \;. \end{cases}$$

Proof. Using the functional equation (1), we can write

$$L\big(2^h + h + a\big) = 2^h + h + a - \log L\big(2^h + h + a\big) \,.$$

Since $L\big(2^h+h\big)=2^h$ by (1) and $\log L$ is a strictly isotone function, we have for $a\in\big[-2^{h-1}-1\mathinner{..}-1\big]$,

$$h-1=\log L\big(2^{h-1}+h-1\big)\leqslant\log L\big(2^h+h+a\big)<\log L\big(2^h+h\big)=h\,,$$

and for $a\in\big[0\mathinner{..}2^h\big]$,

$$h=\log L\big(2^h+h\big)\leqslant\log L\big(2^h+h+a\big)<\log L\big(2^{h+1}+h+1\big)=h+1\,.$$

Thus, the points where $\lceil L\rceil$ does not increase are $\lceil L(2^h+h-1)\rceil=\lceil L(2^h+h)\rceil$. From these observations, the lemma is easily inferred. □

3.1.1 Oscillations

Theorem 3 gives the asymptotic value of $W/2^L$ for parametrization of n 'close' to 2^h+h. See Figure 2 on page 6.

Theorem 3 *Assume that $n\to+\infty$ is of the form $n=2^h+h+a$, where $a=o(2^h)$. Then*

$$\frac{W(n)}{2^{L(n)}}\sim\begin{cases}2\,, & a\leqslant 0\,;\\ 1+2^{-a}\,, & a\geqslant 0\,.\end{cases}$$

Proof. We approximate W and 2^L separately and then consider their ratio. — To estimate $W(n)$, we use (5) and apply Lemma 2. For $a\in\big[-2^{h-1}-1\mathinner{..}-1\big]$,

$$W\left(2^h+h+a\right)=2^{2^h+a+1}+2^{2^{h-1}}+O\Big(2^{2^{h-2}}\Big)\sim 2^{2^h+a+1}+2^{2^{h-1}} \tag{6}$$

as $a=o(2^h)$. Similarly for $a\in\big[0\mathinner{..}2^h\big]$,

$$W\left(2^h+h+a\right)=2^{2^h+a}+2^{2^h}+O\Big(2^{2^{h-1}}\Big)\sim\left(2^a+1\right)2^{2^h}\,. \tag{7}$$

Since $L(n)\sim n\sim 2^h$, expanding L twice using the functional equation (1) yields

$$2^{L(n)}=\frac{2^n}{n-\log L(n)}\sim\frac{2^{2^h+h+a}}{2^h+a}\sim 2^{2^h+a}\,. \tag{8}$$

This leads to the asserted formulae for $W/2^L$: For $a\in\big[-2^{h-1}-1\mathinner{..}-1\big]$,

$$\frac{W}{2^L}\sim\frac{2^{2^h+a+1}+2^{2^{h-1}}}{2^{2^h+a}}=2+2^{-2^{h-1}-a}\sim 2$$

by (6) and (8). For $a\in\big[0\mathinner{..}2^h\big]$,

$$\frac{W}{2^L}\sim\frac{\left(2^a+1\right)2^{2^h}}{2^{2^h+a}}=1+2^{-a}$$

by (7) and (8). □

A complementary view is given by Theorem 4, which describes how $W/2^L$ develops between 2^h+h and $2^{h+1}+h+1$. See Figure 3.

Theorem 4 *Assume that $n\to+\infty$ and write $n=b2^h+h$ with $b\in[1,2]$, $h\in\mathbb{N}$. Then*

$$\frac{W(n)}{2^{L(n)}}\;=\;b\left(1+2^{(1-b)2^h}+O\left(2^{\left(\frac{1}{2}-b\right)2^h}\right)\right)\left(1-\frac{\log b+O(h/n)}{b2^h}\right).$$

In particular, $W(n)/2^{L(n)}\sim b$, if n grows in such a way that b is bounded away from 1.

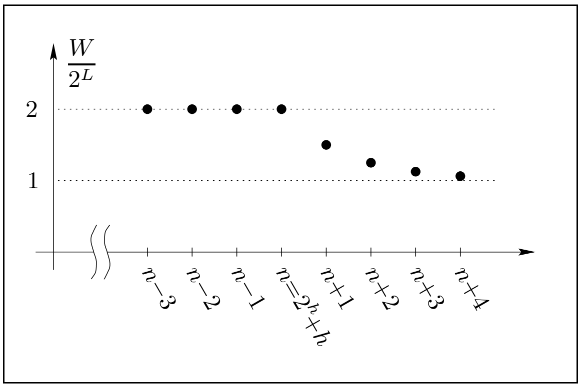

Figure 2: The worst-case size of (q)OBDDs near $n = 2^h + h$

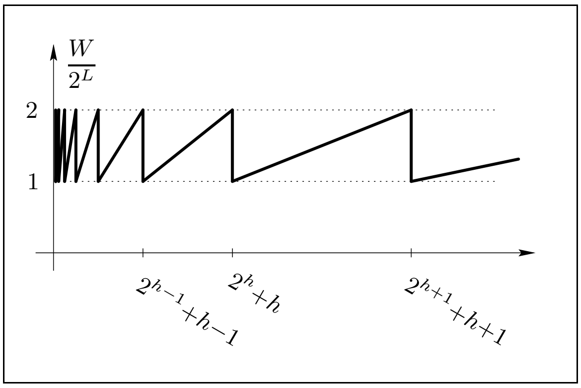

Figure 3: The oscillation of the worst-case size of (q)OBDDs

Proof. It is easy to see that h and b are well-defined for every n. The proof of the theorem is similar to that of the case $a \geqslant 0$ in Theorem 3, but we need to estimate $2^{L(n)}$ more accurately. First, observe that $L(n) = n + O(\log n) = b2^h\bigl(1 + O(h/n)\bigr)$, so from $\log(1 + x) \sim x \log e$ for $x \to 0$ we get

$$\log L(n) = \log\bigl(b2^h(1 + O(h/n))\bigr) = \log b + h + O(h/n)\,.$$

Therefore, a refined version of (8) is

$$2^{L(n)} = \frac{2^{b2^h+h}}{b2^h - \log b + O(h/n)}\,. \tag{8'}$$

Using (7) and (8'), we see that

$$\begin{aligned}\frac{W(n)}{2^{L(n)}} &= \frac{2^{b2^h} + 2^{2^h} + O\Bigl(2^{2^{h-1}}\Bigr)}{2^{b2^h}} \cdot \frac{b2^h - \log b + O(h/n)}{2^h}\\ &= b\,\Bigl(1 + 2^{(1-b)2^h} + O\Bigl(2^{(\frac{1}{2}-b)2^h}\Bigr)\Bigr)\Bigl(1 - \frac{\log b + O(h/n)}{b2^h}\Bigr)\,.\end{aligned}$$

□

3.1.2 Global bounds

Our refined analysis of the oscillation of the worst-case size also leads to slight improvements over the upper bound of Liaw and Lin [11] and Heap and Mercer [9] and the lower bound of Breitbart, Hunt, and Rosenkrantz [1]. Essentially, the lower bound from [1] is $1 + o(1)$, and the upper bound from [11] is $2 + o(1)$. Our bounds are attained for certain sequences of n which are explained in the proof.

Theorem 5 *For $n \to +\infty$,*

$$1 \;<\; \frac{W(n)}{2^{L(n)}} \;\leqslant\; 2 + O\left(2^{-L(n)/2}\right) \;=\; 2 + O\left(\sqrt{\frac{n}{2^n}}\right)\,.$$

The upper bound can be improved to $W(n)/2^{L(n)} < 2$ for large n, unless $n = 2^h + h$ for some $h \in \mathbb{N}$.

Proof. The lower bound holds because $W > 2^{\lceil L\rceil} \geqslant 2^L$ by (5). — For the upper bound, we use the asymptotic from Theorem 4. Let h and b be as defined there. First, we simplify the $O\bigl(2^{(1/2-b)2^h}\bigr)$ term. Observe that

$$L(n) = L\bigl(b2^h + h\bigr) \leqslant L\bigl(b2^h + \log b + h\bigr) = L\bigl(b2^h + \log(b2^h)\bigr) = b2^h$$

by the isotonicity and the defining functional equation of L. Therefore,

$$\left(\frac{1}{2} - b\right) 2^h \leqslant \frac{-b}{2} 2^h \leqslant \frac{-L(n)}{2} \,.$$

This proves

$$2^{\left(\frac{1}{2}-b\right)2^h} = O\bigl(2^{-L(n)/2}\bigr) \,.$$

With $a := (b-1)2^h$ (so $n = 2^h + h + a$) the asymptotic from Theorem 4 becomes

$$\frac{W(n)}{2^{L(n)}} \;=\; \left(1 + \frac{a}{2^h}\right) \left(1 + 2^{-a} + O\bigl(2^{-L(n)/2}\bigr)\right) \left(1 - \frac{\log(1 + a/2^h) + o(1)}{a + 2^h}\right) . \tag{9}$$

In particular, for any sequence of the form $n = 2^h + h + a$, where $a \to +\infty$ and $a = o\bigl(2^h\bigr)$, we get $W/2^L = 1 + o(1)$, which proves that the lower bound is asymptotically tight.

On the other hand, if $n = 2^h + h$, then $L(n) = 2^h$ by Lemma 2, and from (5) we can easily read off that

$$W(2^h + h) = 2 \cdot 2^{2^h} + 2^{2^{h-1}} + O\bigl(2^{2^{h-2}}\bigr) = 2 \cdot 2^{L(n)} + \Theta\bigl(2^{L(n)/2}\bigr) \,,$$

which shows that the upper bound is attained for $a = 0$. (Recall that $L(n) = n - \log n + o(1)$, so $2^L \sim 2^n/n$.)

It remains to show that $W(n)/2^{L(n)} < 2$, if $n = 2^h + h + a$ and $a \in [1 \,..\, 2^h]$, provided h is bigger than some fixed constant (which will not be determined here). In the following estimations, we always assume that h is large enough. Note that $L(n) = \Theta\bigl(2^h\bigr)$. If $a = 1$, then $W/2^L = \frac{3}{2} + o(1) < 2$ by (9). If $2 \leqslant a \leqslant \frac{2}{5} 2^h$, then

$$\Bigl(1 + \underbrace{a/2^h}_{\leqslant 2/5}\Bigr)\bigl(1 + \underbrace{2^{-a}}_{\leqslant 1/4}\bigr) \leqslant \frac{7}{5} \cdot \frac{5}{4} = \frac{7}{4} \,,$$

which implies $W/2^L \leqslant \frac{7}{4} + o(1) < 2$ by (9). Finally, if $\frac{2}{5} 2^h < a \leqslant 2^h$, then

$$\frac{W(n)}{2^{L(n)}} = \underbrace{\left(1 + a/2^h\right)}_{\leqslant 2} \left(1 + \underbrace{2^{-a}}_{\leqslant 2^{-\frac{2}{5}2^h}} + \underbrace{O\bigl(2^{-L(n)/2}\bigr)}_{2^{-\Omega(2^h)}}\right) \left(1 - \underbrace{\frac{\log(1 + a/2^h) + o(1)}{a + 2^h}}_{\geqslant \left(\log \frac{7}{5} + o(1)\right)/2^{h+1}}\right) < 2$$

by (9). ☐

3.2 Reduced OBDDs

We now describe the necessary modifications when we take the effect of the deletion rule into account.

3.2.1 Oscillations

By (4), $j = \lceil i_0 \rceil$ is the smallest index such that $k_j \geqslant m_j$. Since both k_j and m_j are powers of 2 and $m'_j = m_j\bigl(1 - |o(1)|\bigr)$ for $j = n - \omega(1)$, it follows that $j = \lceil i_0 \rceil$ is also the smallest index such that $k_j \geqslant m'_j$. The starting point of our qOBDD analysis was (5). For reduced OBDDs, we have

$$\begin{aligned} W'(n) \;&=\; \sum_{i=1}^{\lceil i_0 \rceil - 1} 2^{i-1} \;+\; \sum_{i=\lceil i_0 \rceil}^{n} \left(2^{2^{n-i+1}} - 2^{2^{n-i}}\right) \\ &=\; \sum_{i=0}^{\lceil L \rceil - 1} 2^i \;+\; \sum_{i=1}^{n - \lceil L \rceil} \left(2^{2^i} - 2^{2^{i-1}}\right) \\ &=\; 2^{\lceil L \rceil} \;+\; 2^{2^{n-\lceil L \rceil}} \;-\; 3 \,. \end{aligned} \tag{10}$$

Theorem 6 *Theorems 3 and 4 hold for* $W'/2^L$ *as well.*

Proof. Since $-3 = O\big(2^{2^{L-2}}\big)$, the estimates (6) and (7) from the proof of Theorem 3 are valid for W', too. □

3.2.2 Global Bounds

The global bound is even nicer for reduced OBDDs.

Theorem 7 *For large enough* n,

$$1 \;<\; \frac{W'(n)}{2^{L(n)}} \;<\; 2\,,$$

and both bounds are asymptotically tight.

Proof. The lower bound follows since $W' > 2^{\lceil L \rceil} \geqslant 2^L$ by (10). —Since $W' \leqslant W$, the upper bound follows from Theorem 5 except for the case $n = 2^h + h$. If $n = 2^h + h$, then $\lceil L \rceil = 2^h$ by Lemma 2, and we have $W' = 2^L + 2^{2^{n-L}} - 3 = 2 \cdot 2^L - 3$ by (10). This also shows that the upper bound is asymptotically attained. □

4 Discussion

The worst-case size is just one way to look at the complexity of Boolean functions. In fact, almost all Boolean functions are close to the worst-case size [10, 11]. Gröpl, Srivastav, and Prömel [6] have shown, building on work of Wegener [15], that the OBDD size of almost all Boolean functions (up to a doubly exponentially small fraction) deviates only by an exponentially small factor from the expected size. Also, the expected size coincides with $W'(n)$ up to a factor of $1 + o(1)$ for most values of n, with the exception of intervals of constant length around the values $n = 2^h + h$. Hence, the expected size oscillates also, but the peaks are slightly smoother. Similar results hold for OBDDs with an optimal variable ordering. However, the exact course of the worst-case OBDD size for optimal variable orderings is still unknown for the exceptional intervals mentioned above.

If we drop the variable ordering restriction, but retain the read-once property, the resulting data structure is called free binary decision diagram (FBDD). FBDDs (with certain restrictions) have efficient algorithms [8, 14]. The expected and worst-case FBDD sizes coincide for most n asymptotically with those of OBDDs and hence oscillate in a similar way, but the exceptional intervals for FBDDs have non-constant length [5]. It seems that no results are known on the worst-case FBDD size except those which follow from the results on BDDs and OBDDs. As a direction of future research, we suggest that the worst-case size of FBDDs should be determined up to a factor of $1 + o(1)$ for all n.

References

[1] Y. Breitbart, H. Hunt III, D. Rosenkrantz: On the Size of Binary Decision Diagrams Representing Boolean Functions; Theoret. Computer Sci., vol. 145, 45 – 69, 1995.

[2] K. S. Brace, R. L. Rudell, R. E. Bryant: Efficient Implementation of a BDD Package; Proc. 27th Design Autom. Conf., 40 – 45, 1990.

[3] R. E. Bryant: Graph-Based Algorithms for Boolean Function Manipulation; IEEE Trans. Computers, vol. C-35, no. 8, 677 – 691, 1986.

[4] R. E. Bryant: Symbolic Boolean Manipulation with Ordered Binary-Decision Diagrams; ACM Computing Surveys, vol. 24, no. 3, 293 – 318, 1992.

[5] C. Gröpl: Binary Decision Diagrams for Random Boolean Functions; Dissertation, Humboldt Universität zu Berlin, available via http://dochost.rz.hu-berlin.de/docserv.html, 126 pages, 1999.

[6] C. Gröpl, A. Srivastav, H. J. Prömel: Size and Structure of Random Ordered Binary Decision Diagrams (Extended Abstract); Proc. 15th Symp. Theoret. Asp. Computer Sci., Springer LNCS 1373, 238 – 248, 1998.

[7] C. Gröpl, A. Srivastav, H. J. Prömel: Size and Structure of Random Ordered Binary Decision Diagrams; preprint, 30 pages, 2000.

[8] J. Gergov, C. Meinel: Efficient Boolean Manipulation with OBDDs can be Extended to FBDDs; IEEE Trans. Computers, vol. 43, no. 10, 1197 – 1209, 1994.

[9] M. A. Heap, M. R. Mercer: Least Upper Bounds on OBDD Sizes; IEEE Trans. Computers, vol. 43, no. 6, 764 – 767, 1994.

[10] C. Y. Lee: Representation of Switching Circuits by Binary Decision Programs; Bell System Technical Journal, vol. 38, 985 – 999, 1959.

[11] Heh-Tyan Liaw, Chen-Shang Lin: On the OBDD-Representation of General Boolean Functions; IEEE Trans. Computers, vol. 41, no. 6, 661 – 664, 1992.

[12] C. E. Shannon: The Synthesis of Two-Terminal Switching Circuits; Bell System Technical Journal, vol. 28, 59 – 98, 1949.

[13] F. Somenzi: CUDD: Colorado University Decision Diagram Package; available via ftp://vlsi.colorado.edu/pub/.

[14] D. Sieling, I. Wegener: Graph Driven BDDs – a New Data Structure for Boolean Functions; Theoret. Computer Sci., vol. 141, 283 – 310, 1995.

[15] I. Wegener: The Size of Reduced OBDDs and Optimal Read-Once Branching Programs for Almost All Boolean Functions; IEEE Trans. Computers, vol. 43, no. 11, 1262 – 1269, 1994.