Representations of Logic Functions using QRMDDs

Shinobu NAGAYAMA[1], Tsutomu SASAO[1,2], Yukihiro IGUCHI[3] and Munehiro MATSUURA[1]
[1]Department of Computer Science and Electronics, Kyushu Institute of Technology
Iizuka 820-8502, Japan
[2]Center of Microelectronics Systems Kyushu Institute of Technology
Iizuka 820-8502, Japan

[3]Department of Computer Science, Meiji University, Kawasaki 214-8571, Japan

Abstract

This paper considers quasi-reduced multi-valued decision diagrams with k bits (QRMDD(k)s) to represent two-valued logic functions. It shows relations between the numbers of nodes in QRMDD(k)s and values of k for benchmark functions; an upper bound on the number of nodes in the QRMDD(k); difference between the upper bound and the number of nodes in the QRMDD(k)s for random functions; and the amount of total memory, evaluation time, and area-time complexity for QRMDD(k)s. Experimental results using standard benchmark functions show that the area-time complexity takes its minimum when k is between 3 and 6.

1. Introduction

With the increase of the complexity of digital systems, representations of logic functions that can evaluate functions efficiently and require small amount of memory are becoming important [2]. In this paper, we consider representations of two-valued logic functions using quasi-reduced multi-valued decision diagrams with k bits (QRMDD(k)s). As for methods to represent logic functions by decision diagrams (DDs), binary decision diagrams (BDDs) [1, 7] and multi-valued decision diagrams (MDDs) [3, 10, 12, 14] are known. Especially, MDDs require fewer nodes than corresponding BDDs. Also, the number of memory accesses required in MDDs is smaller than corresponding BDDs [12]. In this paper, we show relations among the amount of memory to represent QRMDD(k), the number of memory accesses, and values of k.

The rest of the paper is organized as follows:

In Section 2, we will define MDDs and QRMDDs, and explain benchmark functions and random functions.

In Section 3, we obtain an upper bound on the number of nodes in a QRMDD(k). Also, we show an interesting property holds for many of benchmark functions. We also show that random functions do not have this property.

In Section 4, we introduce the measure called *area-time complexity*. When we use a QRMDD(k), the number of memory accesses decreases with k, while the amount of memory to represent it increase with k. We are interested in k that reduces the number of memory accesses without increasing the amount of memory excessively. To obtain such k, we introduce a measure called *area-time complexity*. By experiments, the measure is the minimum when k is between 3 and 6.

2 Definitions

This section defines quasi-reduced multi-valued decision diagrams(QRMDDs), shows a method to represent multiple-output functions, and introduces benchmark functions.

2.1 Representation of Logic Functions

Let $f(X)$ be a two-valued logic function, where $X = (x_1, x_2, \ldots, x_n)$. Let $x_i (i = 1, 2, \ldots, n)$ be binary variables, where $n = rk$.

Let $X = (X_1, X_2, \ldots, X_r)$ be a partition of X, where $\{X\} = \{X_1\} \cup \{X_2\} \cup \ldots \cup \{X_r\}$ and $\{X_i\} \cap \{X_j\} = \phi (i \neq j)$, and each X_j consists of k binary variables. Then, a two-valued logic function $f(X)$ can be represented by $f(X_1, X_2, \ldots, X_r)$: $\{0, 1, 2, \ldots, 2^k - 1\}^r \rightarrow B$, where $B = \{0, 1\}$.

2.2 QRMDD

As for the definitions on MDDs, refer [10].

Definition 2.1 *The reduced ordered multi-valued decision diagram (RMDD) having non-terminal nodes with 2^k edges is denoted by RMDD(k). Especially, when $k = 1$, RMDD(1) is a* **reduced ordered binary decision diagram** *(RBDD).*

Definition 2.2 *In a decision diagram (DD), a path from the root node to a terminal node is a* **path of DD**. *The number of edges on the path is the* **length of the path**.

Definition 2.3 *In a DD, the* **number of nodes in the DD**, *denoted by $nodes(DD)$, includes terminal nodes.*

Definition 2.4 *When all X_i $(i = 1, 2, \ldots, u)$ appear in this order on an arbitrary path of an MDD(k), the MDD(k) is a* **quasi-reduced multi-valued decision diagram with k bits** *(QRMDD(k)).*

The length of an arbitrary path in a QRMDD(k) is equal to u, the number of input variables. An RMDD has no redundant nodes, while a QRMDD usually has redundant nodes. Therefore, we have the following relation in the number of nodes of an RMDD and its corresponding QR-MDD:

$$nodes(RMDD(k)) \leq nodes(QRMDD(k)).$$

In general, QRMDDs (QRBDDs) require more nodes than corresponding RMDDs (RBDDs). However, QR-MDDs (QRBDDs) have the following features:

1. When the RBDDs are too large to be store in the main memory, QRBDDs can be processed much faster than corresponding RBDDs [15].
2. QRBDDs can be manipulated efficiently by parallel processors [16].
3. QRBDDs (QRMDDs) are used to design LUT cascades [22].

The most severe limitation of conventional BDDs is their size. When a BDD does not fit in the main memory, the BDD must uses the secondary memory. This will increase the number of page faults, and access to the secondary memory [24]. In such a case, quasi-reduced decision diagrams can be used to reduce the page faults. This approch is useful when the quasi-reduced decision diagram is not so greater than the corresponding reduced decision diagrams.

Definition 2.5 *Let the input variables be $X = (X_1, X_2, \ldots, X_u)$. Consider a QRMDD for a function $f(X)$. The number of non-terminal nodes in the QRMDD with respect to a variable X_i is* **width of the QRMDD with respect to** X_i*, and denoted by $width(QRMDD(k), i)$.*

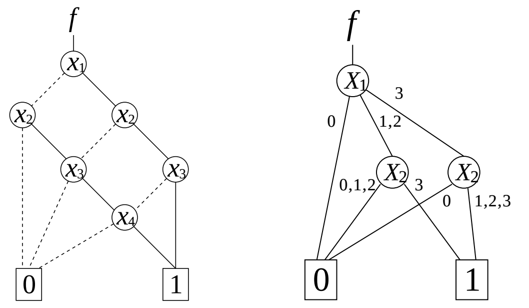
The total number of nodes in the QRMDD(k) is given by

$$nodes(QRMDD(k)) = \sum_{i=1}^{u} width(QRMDD(k), i).$$

Example 2.1 *Consider the function:*

$$f = x_1x_2x_3 \vee x_2x_3x_4 \vee x_3x_4x_1 \vee x_4x_1x_2.$$

The RBDD, the RMDD(2), and the QRMDD(2) for f are shown in Fig. 2.1(a), Fig. 2.1(b), and Fig. 2.2, respectively. In Fig. 2.1(a), the solid lines and the broken lines denote 1-edges and 0-edges, respectively. In Fig. 2.1(b), the input variables $X = (x_1, x_2, x_3, x_4)$ are partitioned into $X = (X_1, X_2)$, where $X_1 = (x_1, x_2)$ and $X_2 = (x_3, x_4)$. We have $nodes(RBDD) = 8$, $nodes(RMDD(2)) = 5$, and $nodes(QRMDD(2)) = 6$. Note that $width(QRMDD(2), 2)$ = 3. (End of Example)



(a) **RBDD** (b) **RMDD(2)**

Figure 2.1. DDs.

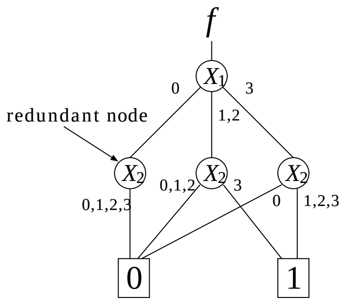


Figure 2.2. QRMDD(2).

2.3 Representations of Multiple-Output Functions

Logic networks usually have many outputs. In most cases, independent representation of each output is inefficient. Let the multiple-output functions be $F = (f_0, f_1, \ldots, f_{m-1})$: $B^n \rightarrow B^m$, where $B = \{1, 0\}$, and n and m denote the number of inputs and outputs, respectively. Several methods exist to represent multiple-output functions by using BDDs [13, 18, 19, 20]. In this paper, we use an encoded characteristic function for non-zero output (ECFN) [21] to represent multiple-output functions. In the following, an RBDD means a BDD for an ECFN, and we assume that RMDDs and QRMDDs are generated from these RBDDs.

2.4 Benchmark Functions

In this paper, we use 131 benchmark functions [6, 25] shown in Table 2.1, where n and m denote the number of inputs and outputs, respectively. In this table, the benchmark functions under $sequential$ originally represent sequential circuits. We removed flip-flops(FFs) from sequential circuits to make them combinational. Such functions are renamed by appending 'c' to the original names. Encodings of BDDs for ECFNs and input variable orderings are obtained in [22]. Details of experimental results are omitted due to the page limitation.

Table 2.1. Benchmark Functions.

$Name$	n	m	$Name$	n	m	$Name$	n	m
$C432$	36	7	$frg1$	28	3	$soar$	83	94
$C499$	41	32	$frg2$	143	139	$spla$	16	46
$C880$	60	26	$i1$	25	16	$t1$	21	23
$C1355$	41	32	$i2$	201	1	$t2$	17	16
$C1908$	33	25	$i3$	132	6	$table5$	17	15
$C2670$	233	140	$i4$	192	6	$tcon$	17	16
$C3540$	50	22	$i5$	133	66	$term1$	34	10
$C5315$	178	123	$i6$	138	67	ti	47	72
$C7552$	207	108	$i7$	199	67	too_large	38	3
$accpla$	50	69	$i8$	133	81	$ts10$	22	16
$al2$	16	47	$i9$	88	63	$ttt2$	24	21
$alcom$	15	38	$i10$	257	224	$unreg$	36	16
$apex1$	45	45	ibm	48	17	vda	17	39
$apex2$	39	3	$in1$	16	17	$vg2$	25	8
$apex3$	54	50	$in2$	19	10	$vtx1$	27	6
$apex5$	117	88	$in3$	35	29	$x1$	51	35
$apex6$	135	99	$in4$	32	20	$x3$	135	99
$apex7$	49	37	$in5$	24	14	$x4$	94	71
$b2$	16	17	$in6$	33	23	$x1dn$	27	6
$b3$	32	20	$in7$	26	10	$x2dn$	82	56
$b4$	33	23	jbp	36	57	$x6dn$	39	5
$b9$	41	21	$k2$	45	45	$x7dn$	66	15
$bc0$	26	11	lal	26	19	$x9dn$	27	7
bca	26	46	$mainpla$	27	54	$xparc$	41	73
bcb	26	39	$mark1$	20	31	$sequential$		
bcc	26	45	$misex2$	25	18	$s208_c$	18	9
bcd	26	38	$misg$	56	23	$s298_c$	17	20
$c8$	28	18	$mish$	94	43	$s344_c$	24	26
cc	21	20	$misj$	35	14	$s349_c$	24	26
$chkn$	29	7	$mlp10$	20	20	$s382_c$	24	27
cht	47	36	mux	21	1	$s400_c$	24	27
$cm150a$	21	1	my_adder	33	17	$s420_c$	34	17
$comp$	32	3	opa	17	69	$s444_c$	24	27
$cordic$	23	2	$pair$	173	137	$s510_c$	25	13
$count$	35	16	$pcle$	19	9	$s526_c$	24	27
cps	24	109	$pcler8$	27	17	$s641_c$	54	43
$dalu$	75	16	pdc	16	40	$s713_c$	54	42
des	256	245	$pm1$	16	13	$s820_c$	23	24
$dk48$	15	17	$rckl$	32	7	$s832_c$	23	24
$duke2$	22	29	rot	135	107	$s838_c$	66	33
$e64$	65	65	sct	19	15	$s1196_c$	32	32
$ex4$	128	28	seq	41	35	$s1423_c$	91	79
$example2$	85	66	$shift$	19	16	$s5378_c$	214	228
$exep$	30	63	$signet$	39	8	$s9234_c$	247	250

3 Number of Nodes in QRMDD(k)

In this part, we first obtain an upper bound on the number of nodes in a QRMDD(k). Then, we obtain the sizes of QRMDD(k)s for benchmark functions, and show that an interesting property holds for many of them. Finally, we obtain the sizes of QRMDD(k)s for randomly generated logic functions, and show that they can be estimated by the upper bound.

3.1 General Functions

For arbitrary logic functions, we have the following:

Theorem 3.1 *An arbitrary n-input logic function can be represented by a QRBDD with at most*

$$2^{n-r} - 1 + \sum_{i=0}^{r} 2^{2^i}$$

nodes, where r is the largest integer that satisfies relation

$$n - r \geq 2^r.$$

Theorem 3.2 *An arbitrary n-input logic function can be represented by a QRMDD(k) with at most*

$$\frac{2^{sk}-1}{2^k-1} + \sum_{i=0}^{t-1} 2^{2^{n-(s+i)k}} + 2$$

nodes, where s and t are the smallest integer that satisfy relations

$$s \geq \frac{n-r}{k}, \quad t \geq \frac{n}{k} - s.$$

The proofs of Theorem 3.1 and Theorem 3.2 are shown in Appendix.

3.2 Benchmark Functions

For each benchmark function in Table 2.1, we counted the number of nodes in QRMDD(k)s for different k. In Table 3.1, ave denotes arithmetic average of the relative sizes, where the number of nodes in QRMDD(1) is set to 1.00, and $stdv$ denotes the standard deviation. We consider the following:

$$\eta = \left| 1 - \frac{k \cdot nodes(QRMDD(k))}{nodes(QRMDD(1))} \right|.$$

Since 119 functions out of 131 functions in Table 2.1 satisfy the relation $\eta < 0.1$, we have

Property 3.1

$$nodes(QRMDD(k)) \simeq \frac{1}{k} nodes(QRMDD(1))$$

Property 3.1 holds for the 119 functions in Table 2.1. As for remaining 12 functions, $\eta \geq 0.1$ holds. Table 3.2 lists these 12 functions. Note that Table 2.1 does not contain functions having small inputs and outputs.

Table 3.1. Relation of nodes in QRMDD(k) and k for benchmark functions.

	k				
	1	2	3	4	5
ave	1.00	0.50	0.33	0.25	0.20
$stdv$	0.000	0.014	0.007	0.013	0.009

Table 3.2. functions with $\eta \geq 0.1$

$Name$	$Circuit$	$Name$	$Circuit$
$C499$	$error\_correcting$	my_adder	$adder$
$C1908$	$error\_correcting$	$pcle$	$control$
$comp$	$comparator$	$tcon$	$control$
$i3$	$control$	$vg2$	$control$
$in1$	$control$	$vtx1$	$control$
$mlp10$	$multiplier$	$x1dn$	$control$

Table 3.3. The number of nodes in QRMDD(k) for random functions.

	k				
n	1	2	3	4	5
10	249.4	103.0	79.0	35.0	35.0
11	439.1	253.2	91.0	181.2	39.0
12	756.0	358.5	298.5	274.5	51.0
13	1294.8	598.6	589.2	279.0	286.6
14	2318.0	1376.1	603.0	291.0	1052.1
15	4343.1	1627.0	843.0	531.0	1059.0
16	8338.5	5348.5	4556.5	4240.5	1063.0
17	16167.3	5723.0	4699.0	4375.0	1075.0
18	31157.9	19975.9	4939.0	4387.0	1315.0
19	58838.4	22107.0	30480.4	4627.0	26852.4
20	107222.3	63272.3	37467.0	45780.3	33827.0

Table 3.4. Upper bound on the number of nodes in QRMDD(k)s.

	k				
n	1	2	3	4	5
10	277	103	79	35	35
11	533	347	91	275	39
12	789	359	331	275	51
13	1301	603	591	279	291
14	2325	1383	603	291	1060
15	4373	1627	843	531	1059
16	8469	5479	4687	4371	1063
17	16661	5723	4699	4375	1075
18	33045	21863	4939	4387	1315
19	65813	22107	37455	4627	33828
20	131349	87399	37467	69907	33827

3.3 Random Functions

We examined whether Property 3.1 holds for random functions. For each n, we randomly generated 2^{n-1} minterms. Table 3.3 shows average numbers of nodes in QRMDD(k)s for n-input random functions. For each n, we generated 10 samples. The deviation of each data is within $\pm 2\%$ of the averages. Table 3.4 shows upper bounds on the numbers of nodes in QRMDD(k)s derived from Theorem 3.2.

The ratio of difference γ between upper bounds and experimental results on the number of nodes in QRMDD(1) for n-input random functions is computed as follows:

$$\gamma = \frac{upper\ bound - experimental\ result}{upper\ bound} \times 100$$

Table 3.5 shows that γ is large after r changes, and is small in other cases. This means that the size of QRMDD(k) for randomly generated functions can be estimated by Theorem 3.2. Table 3.3 also shows that Property 3.1 doesn't hold for random functions. This fact shows that benchmark functions have quite different property from random functions.

Table 3.5. Ratio of difference γ for random functions.

n	r	$\gamma(\%)$	n	r	$\gamma(\%)$	n	r	$\gamma(\%)$
5	1	25.24	12	3	4.81	19	3	10.60
6	2	28.11	13	3	0.48	20	4	18.37
7	2	16.42	14	3	0.30	21	4	4.50
8	2	12.00	15	3	0.68	22	4	0.37
9	2	11.07	16	3	1.54	23	4	0.00
10	2	9.96	17	3	2.96	24	4	0.00
11	3	17.62	18	3	5.71	25	4	0.01

4 Area-Time Complexity of QRMDD(k)

When we use QRMDD(k), the amount of memory access decreases with k, while the total amount of memory increases with k. Thus, we are interested in finding k that reduces the number of memory access without increasing the total amount of memory excessively. To obtain such k, we introduce a measure called *area-time complexity*.

4.1 Model of Computation

We assume the followings:

1. MDDs are implemented directly, not simulated by using BDD packages [10].
2. Encoded input values are available as inputs, and their access time is negligible. For example, when $X_1 = (x_1, x_2, x_3, x_4) = (1, 0, 0, 1)$, $X_1 = 9$ is available as an input.
3. Access to the MDD nodes is time-consuming.

In this case, the computation time is proportional to the number of access to the MDD nodes.

4.2 Amount of Memory to Represent QRMDD(k)

Because QRMDD(k) evaluates variables $X_1, X_2, \ldots, X_u$ in this order, we can use a counter to obtain the next variable to evaluate. Therefore, when a QRMDD(k) is stored in a memory, we need not store an index in a node, but have only to store the next addresses. On the other hand, in an RMDD(k), we have to store an index and the next addresses because the next variable to evaluate may be different in different paths.

Example 4.1 *Fig.* 4.1 *and Fig.* 4.2 *illustrate data structures of a node in a QRMDD(*2*) and an RMDD(*2*), respectively.* *(End of Example)*

Because each node in a QRMDD(k) has 2^k edges, we need

$$2^k nodes(QRMDD(k))$$

words to represent all nodes in a QRMDD(k). When a DD is stored in a memory, each node requires a unique address. The number of bits necessary to specify the address of a node is

$$\lceil log_2 nodes(DD) \rceil .$$

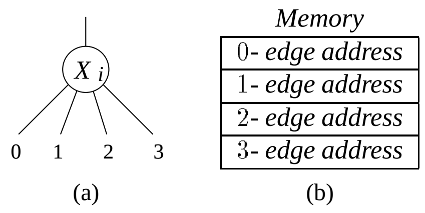


Figure 4.1. Data structure of a node in QRMDD(2).

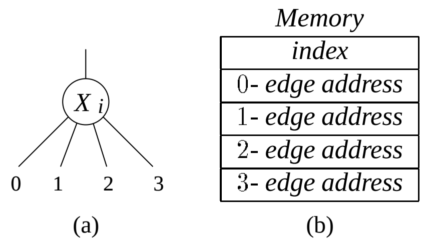


Figure 4.2. Data structure of a node in RMDD(2).

Therefore, the total amount of memory to represent a QRMDD(k) is

$$2^k nodes(QRMDD(k))\lceil log_2 nodes(QRMDD(k))\rceil.$$

4.3 Area-Time Complexity of QRMDD(k)s

Because a QRMDD(k) evaluates k variables at a time, the number of memory accesses of a QRMDD(k) is $\frac{1}{k}$ of the corresponding QRMDD(1). On the other hand, the amount of memory necessary to store a QRMDD(k) node increases with 2^k. In this section, we consider the area-time complexity [5, 23] for QRMDD(k) and obtain the k that minimizes the area-time complexity.

Definition 4.1 *The* **area-time complexity** *is the measure of computational cost considering both area and time. It is defined by*

$$AT = (area) \times (time),$$

or

$$AT^2 = (area) \times (time)^2.$$

Table 4.1. Relation of k and A for QRMDD(k) for benchmark functions.

	k				
	1	2	3	4	5
ave	1.00	0.91	1.14	1.65	2.54
$stdv$	0.000	0.036	0.070	0.114	0.190

Table 4.2. Relation of k and AT for QRMDD(k) for benchmark functions.

	k				
	1	2	3	4	5
ave	1.00	0.46	0.39	0.43	0.53
$stdv$	0.000	0.019	0.027	0.032	0.047

Table 4.3. Relation of k and AT^2 for QRMDD(k) for benchmark functions.

	k						
	1	2	3	4	5	6	7
ave	1.000	0.233	0.133	0.112	0.113	0.131	0.166
$stdv$	0.000	0.011	0.011	0.011	0.014	0.023	0.029

In this paper, the area corresponds to the necessary amount of memory to represent a QRMDD(k), and the time corresponds to the number of memory accesses to evaluate it.

The measure AT is used when both the amount of memory and the number of memory accesses are equally important. On the other hand, the measure AT^2 is used when the number of memory accesses is more important than the amount of memory. For example, AT can be used for embedded systems [2], while AT^2 can be used for logic simulators [8, 9].

4.4 Experimental Results

For each benchmark function in Table 2.1, we obtained three measures A, AT, and AT^2. Table 4.1, Table 4.2, and Table 4.3 show the relation of k and A, AT, and AT^2, respectively. In these tables, ave denotes the arithmetic average, and $stdv$ denotes the standard deviation for benchmark functions.

For each benchmark function in Table 2.1, A takes its minimum when $k = 2$; AT takes its minimum when $k = 3$ or $k = 4$; and AT^2 takes its minimum when $k = 4 \sim 6$.

4.5 Analysis for the Functions that Satisfy Property 3.1

In Section 4.4, for QRMDD(k)s, we found values of k that make A, AT and AT^2 minimum, experimentally. In this section, we assume that Property 3.1 holds, and will find the values k that make A, AT and AT^2 minimum, analytically. Let A and T be the necessary amount of memory to represent a QRMDD(k), and the number of memory accesses necessary to evaluate a QRMDD(k), respectively. Then, we have the following:

$$A = 2^k nodes(QRMDD(k))\lceil log_2 nodes(QRMDD(k))\rceil,$$

$$T = \lceil \frac{n}{k} \rceil.$$

Let assume that Property 3.1 holds and $nodes(QRMDD(1)) = N$. Then we have:

$$A \simeq \frac{2^k}{k} N \lceil log_2 \frac{N}{k} \rceil,$$

$$AT \simeq \frac{2^k n}{k^2} N \lceil log_2 \frac{N}{k} \rceil,$$

and

$$AT^2 \simeq \frac{2^k n^2}{k^3} N \lceil log_2 \frac{N}{k} \rceil.$$

Note that N is usually greater than a few hundreds, while k is usually at most 7. Thus, we can use the following approximation:

$$\lceil log_2 N - log_2 k \rceil \simeq \lceil log_2 N \rceil.$$

Therefore, A, AT and AT^2 can be simplified to

$$A \simeq \frac{2^k}{k} C_0, \quad AT \simeq \frac{2^k}{k^2} C_1, \quad and \quad AT^2 \simeq \frac{2^k}{k^3} C_2,$$

respectively, where the constants C_0, C_1 and C_2 are independent of k. From the above formulas, we can see that A, AT and AT^2 take their minimum when $k = 2$, $k = 3$ and $k = 4$, respectively.

5 Conclusion and Comments

In this paper, we considered a method to represent logic functions by using QRMDD(k)s. Especially, 1) We derived an upper bound on the number of nodes in a QRMDD(k), and showed that the numbers of nodes in QRMDD(k)s for random functions can be estimated by the bound, 2) We showed that Property 3.1 holds for many benchmark functions, and 3) We showed that the area-time complexity for QRMDD(k) takes its minimum when $k = 3 \sim 6$, and 4) We showed that benchmark functions have quite different property from randomly generated functions.

Acknowledgments

This research is partly supported by the Grant in Aid for Scientific Research of The Japan Society for the Promotion of Science (JSPS), and the Takeda Foundation.

References

[1] P. Ashar and S. Malik, "Fast functional simulation using branching programs," *ICCAD'95*, pp. 408–412, Nov. 1995.

[2] F. Balarin, M. Chiodo, P. Giusto, H. Hsieh, A. Jurecska, L. Lavagno, A. Sangiovanni-Vincentelli, E. M. Sentovich, and K. Suzuki, "Synthesis of software programs for embedded control applications," *IEEE Trans. CAD*, Vol. 18, No. 6, pp.834-849, June 1999.

[3] B. Becker and R. Drechsler, "Efficient graph based representation of multi-valued functions with an application to genetic algorithms," *Proc. of International Symposium on Multiple Valued Logic*, pp. 40-45, May 1994.

[4] R. K. Brayton, "The future of logic synthesis and verification," in S. Hassoun and T. Sasao (eds.) *Logic Synthesis and Verification*, Kluwer Academic Publisher, 2001 pp. 408-434.

[5] R. P. Brent and H. T. Kung, "The area-time complexity of binary multiplication," *Journal of the ACM*, Vol. 28, No. 3, pp. 521-534, July 1981.

[6] F. Brglez and H. Fujiwara, "Neutral netlist of ten combinational benchmark circuits and a target translator in FORTRAN," *Special session on ATPG and fault simulation, Proc. IEEE Int. Symp. Circuits and Systems*, June 1985, pp. 663-698.

[7] R. E. Bryant, "Graph-based algorithms for boolean function manipulation," *IEEE Trans. Comput.*, Vol. C-35, No. 8, pp. 677–691, Aug. 1986.

[8] Y. Iguchi, T. Sasao, M. Matsuura, and A. Iseno "A hardware simulation engine based on decision diagrams," *Asia and South Pacific Design Automation Conference (ASP-DAC'2000)*, Yokohama, Japan, Jan. 26-28, 2000, pp. 73-76.

[9] Y. Iguchi, T. Sasao, M. Matsuura, "Implementation of multiple-output functions using PQMDDs," *Proc. of International Symposium on Multiple-Valued Logic*, pp. 199-205, May 2000.

[10] T. Kam, T. Villa, R. K. Brayton, and A. L. Sangiovanni-Vincentelli, "Multi-valued decision diagrams: Theory and Applications," *Multiple-Valued Logic*, 1988, Vol. 4, No. 1-2, pp. 9–62, 1998.

[11] H.-T. Liaw, and C.-S. Lin. "On the OBDD-representation of general Boolean function," *IEEE Transactions on Computers*, Vol. 4, No. 6, pp. 661–664, June 1992.

[12] P. C.McGeer, K. L. McMillan, A. Saldanha, A. L. Sangiovanni-Vincentelli, and P. Scaglia, "Fast discrete function evaluation using decision diagrams," *ICCAD'95*, pp. 402–407, Nov. 1995.

[13] S. Minato, N. Ishiura, and S. Yajima, "Shared binary decision diagram with attributed edges for efficient Boolean function manipulation," *Proc. 27th ACM/IEEE Design Automation Conf.*, pp. 52–57, June 1990.

[14] D. M Miller, "Multiple-valued logic design tools," *Proc. of International Symposium on Multiple Valued Logic*, pp. 2–11, May 1993.

[15] H. Ochi, K. Yasuoka and S. Yajima, "Breadth-first manipulation of very large binary-decision diagrams," *1993 IEEE/ACM International Conference on Computer-Aided Design*, pp. 48-55, Nov. 1993.

[16] H. Ochi, N. Ishiura and S. Yajima, "Breadth-first manipulation of SBDD of Boolean function for vector processing," *28th ACM/IEEE Design Automation Conference*, pp. 413-416, 1991.

[17] T. Sasao, "FPGA design by generalized functional decomposition," (Sasao ed.) *Logic Synthesis and Optimization*, Kluwer Academic Publishers, 1993, pp. 233-258.

[18] T. Sasao and M. Fujita (eds.), *Representations of Discrete Functions*, Kluwer Academic Publishers 1996.

[19] T. Sasao and J. T. Butler, "A method to represent multiple-output switching functions by using multi-valued decision diagrams," *Proc. of International Symposium on Multiple-Valued Logic*, pp. 248-254, Santiago de Compostela, Spain, May 29-31, 1996.

[20] T. Sasao, *Switching Theory for Logic Synthesis*, Kluwer Academic Publishers, 1999.

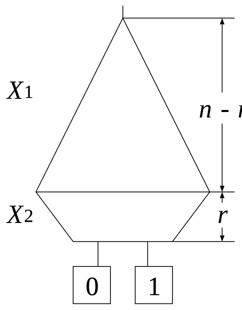


Figure A.1. Partition of BDD.

[21] T. Sasao, "Compact SOP representations for multiple-output functions: An encoding method using multiple-valued logic," *Proc. of International Symposium on Multiple-Valued Logic*, Warsaw, Poland, May 22 - 24, 2001, pp. 207-211.

[22] T. Sasao, M. Matsuura, and Y. Iguchi, "Cascade realization of multiple-output function and its application to reconfigurable hardware," *International Workshop on Logic and Synthesis*, Lake Tahoe, June 2001, pp. 225-230.

[23] C. D. Thompson, "Area-Time Complexity for VLSI," *Ann. Symp. on Theory of Computing*, May 1979 pp. 81-89.

[24] I. Wegener, *Branching Programs and Binary Decision Diagrams: Theory and Applications*, SIAM 200.

[25] S. Yang, *Logic synthesis and optimization benchmark user guide version 3.0*, MCNC, Jan. 1991.

A Appendix

A.1 Proof of Theorem 3.1

Definition A.1 *Suppose that a quasi-reduced ordered binary decision diagram (QRBDD) for an n-input logic function is partitioned into two parts as shown in Fig $A.1$. It is partitioned into* **the upper part** *and* **the lower part**. *The upper part has the variables $X_1 = (x_1, x_2, \ldots, x_{n-r})$, while the lower part has the variables $X_2 = (x_{n-r+1}, \ldots, x_n)$.*

Proof of Theorem 3.1 The number of non-terminal nodes in the complete binary tree for n variables is $2^n - 1$. This gives the number of nodes in the upper part. As for the lower part, the width of QRBDD becomes narrow because the nodes representing the same functions are deleted. The upper bound on the number of nodes in a BDD becomes minimum when r is the maximum integer satisfying $n - r \geq 2^r$ [11]. As for the lower part, at most 2^{2^r} r-input logic functions are represented. When all of these functions are represented, the number of nodes in the lower part becomes maximum. In this case, the BDD represents the logic function as follows:

$$f(X_1, X_2) = \bigvee_{\vec{a}_i \in B^{n-r}} X_1^{\vec{a}_i} f(\vec{a}_i, X_2),$$

where

$$X_1^{\vec{a}_i} = \begin{cases} 1 & (X_1 = \vec{a}_i) \\ 0 & (otherwise). \end{cases}$$

The upper part realizes $X_1^{\vec{a}_i}$, and the lower part realizes $f(\vec{a}_i, X_2)$. Because $f(\vec{a}_i, X_2)$ is an r-input logic function, at most 2^{2^r} different $f(\vec{a}_i, X_2)$ exist in the lower part. When 2^{2^i} functions are realized for each i $(i = 0, \ldots, r)$ from the terminal node to the r, the QRBDD has the maximum number of nodes. Therefore, the number of nodes in a QRBDD is at most

$$2^{n-r} - 1 + \sum_{i=0}^{r} 2^{2^i}.$$

(Q.E.D)

A.2 Proof of Theorem 3.2

First, we will consider Lemma A.1 to prove Theorem 3.2.

Lemma A.1 *Let an n-input logic function be $f(X)$, where $X = (x_1, x_2, \ldots, x_n)$. Let $f(X)$ be decomposed as*

$$f(X) = g_i(h(X_1, X_2, \ldots, X_i), X_{i+1}, \ldots, X_u),$$

where $u = \lceil \frac{n}{k} \rceil$, and let μ_i be the column multiplicity of the decomposition for $i = 1, 2, \ldots, u$. Then, the number of nodes in the QRMDD(k) for f is given by

$$nodes(QRMDD(k)) = \sum_{i=1}^{u} \mu_i.$$

Proof Since the width of the QRMDD(k) with respect to a variable X_i is equal to the column multiplicity μ_i [17, 18], we have the lemma. (Q.E.D)

Proof of Theorem 3.2 By Lemma A.1, the number of nodes in a QRMDD(k) for an n-input logic function is equal to the sum of the column multiplicities that are obtained by iterative functional decompositions. The upper part of the QRBDD is decomposed into s parts. In this case, $\mu_1, \mu_2, \ldots, \mu_s$ are $1, 2^k, 2^{2k}, \ldots, 2^{sk}$. It is a geometric progression with the initial term 1 and the common ratio 2^k. The lower part of the QRBDD is decomposed into t parts. Also in the lower part, the number of nodes is equal to the sum of the column multiplicities. In other words, it is equal to the sum of the widths of a QRBDD with respect to corresponding variables. When $n \neq (s+t)k$, the number of variables in X_u is not a multiple of k. So, the column multiplicity for this part is computed separately. Note that $\mu_u = 2$, since f is a two-valued logic function. Thus, the number of nodes in QRMDD(k) is at most

$$\frac{2^{sk} - 1}{2^k - 1} + \sum_{i=0}^{t-1} 2^{2^{n-(s+i)k}} + 2.$$

(Q.E.D)