

QRMDD を用いた論理関数の表現法について

永山 忍¹ 笹尾 勤^{2,3} 井口 幸洋¹ 松浦 宗寛²

¹ 明治大学 理工学部

² 九州工業大学 情報工学部

³ 九州工業大学 マイクロ化総合技術センタ

あらまし: 本論文では, QRMDD(k)(Quasi-Reduced Multi-valued Decision Diagram with k bits) を用いた論理関数の表現に関して, 以下のことを示す. ベンチマーク関数における, QRMDD(k) の節点数と k の値の関係. また QRMDD(k) の節点数の上界および, その上界と乱数関数を QRMDD(k) で表現したときの節点数の差. QRMDD(k) の総メモリ量, 評価時間, および面積時間複雑度. $k = 3 \sim 6$ の時, 面積時間複雑度が最小となること.

和文キーワード: QRMDD, 面積時間複雑度, 乱数関数.

Representations of Logic Functions using QRMDDs

Shinobu NAGAYAMA¹, Tsutomu SASAO^{2,3}, Yukihiro IGUCHI¹
and Munehiro MATSUURA²

¹ Department of Computer Science, Meiji University

² Department of Computer Science and Electronics, Kyushu Institute of Technology

³ Center of Microelectronic Systems, Kyushu Institute of Technology

Abstract: In this paper, we consider quasi-reduced multi-valued decision diagrams with k bits (QRMDD(k)s) to represent logic functions. We will show relations between the number 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.

Key words: QRMDD, area-time complexity, random functions.

1 はじめに

近年、デジタルシステムが大規模化したため、評価が高速で、かつメモリ量が少ない論理関数の表現法が重要となってきている [2]. 本論文では、QRMDD(Quasi-Reduced Multi-valued Decision Diagram) を用いた論理関数の表現法について考察を行なう. 決定図 (DD: Decision Diagram) を用いた論理関数の表現法として、BDD(Binary Decision Diagram) [1, 5] や MDD(Multi-valued Decision Diagram) [3, 8, 10, 12] が知られている. 特に MDD は、節点数を少なくできる. またメモリアクセス回数を BDD の場合の k 分の 1 に削減できる [10]. 本論文では、 $QRMDD(k)$ を表現するためのメモリ量やメモリアクセス回数と k の値との関係を示す.

2 諸定義

本節では、QRMDD(Quasi-Reduced Multi-valued Decision Diagram) を定義し、多出力関数の表現法、および本論文の実験に用いたベンチマーク関数を示す.

2.1 QRMDD

定義 2.1 各非終端節点が 2^k 個の枝を持つ *RMDD* (*Reduced Multi-valued Decision Diagram*) を $RMDD(k)$ と表記する. 特に $k = 1$ のとき、 $RMDD(1)$ は *RBD* (*Reduced ordered Binary Decision Diagram*) を表す.

定義 2.2 *DD* (*Decision Diagram*) において、根から終端節点までの経路を *DD* のパスという. このときパス上の枝数をパス長という.

定義 2.3 *DD* において、終端節点を含んだ節点の総数を *DD* の節点数といい、 $nodes(DD)$ と表記する.

定義 2.4 $MDD(k)$ の任意のパス上に、全ての X_i ($i = 1, 2, \dots, u$) がこの順に現れるとき、この $MDD(k)$ を $QRMDD(k)$ という.

$QRMDD(k)$ の任意のパスのパス長は入力変数の個数に等しい. $RMDD$ は冗長な節点を含まない MDD であり、 $QRMDD$ は $RMDD$ に冗長な節点を付け加えた MDD なので、 $RMDD$ と $QRMDD$ の節点数に関して、関係

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

が成立する.

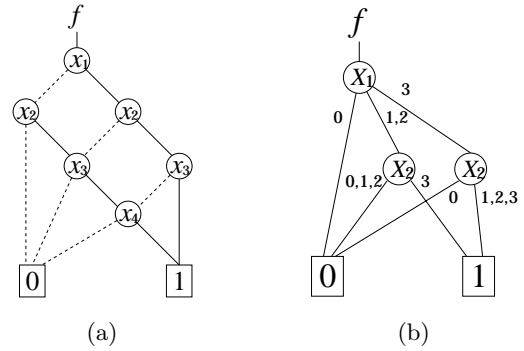


図 2.1: DD の例

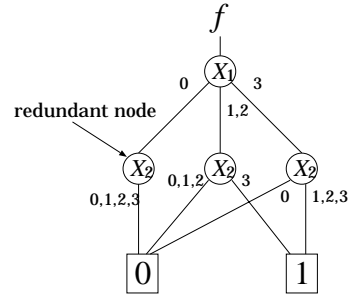


図 2.2: QRMDD(2) の例

定義 2.5 入力変数を $X = (X_1, X_2, \dots, X_u)$ とする. 関数 $f(X)$ を表現する $QRMDD$ を考える. 変数 X_i における $QRMDD$ の節点数を $QRMDD$ の X_i における幅といい、 $width(QRMDD(k), i)$ で表す.

$QRMDD(k)$ の節点数は、

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

で計算できる.

例 2.1 関数

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

を表現する *RBDD* を図 2.1(a) に、 $RMDD(2)$ を図 2.1(b) に、 $QRMDD(2)$ を図 2.2 に示す. 図 2.1(a) の *RBD* で実線は 1-枝、破線は 0-枝を示す. また、図 2.1(b) の $RMDD$ で $X_1 = (x_1, x_2)$, $X_2 = (x_3, x_4)$ である. $nodes(RBDD) = 8$, $nodes(RMDD(2)) = 5$, そして $nodes(QRMDD(2)) = 6$ である. また $width(QRMDD(2), 2) = 3$ である. (例終り)

2.2 多出力関数の表現

論理回路は、通常多出力である。各出力を独立に表現すると、ほとんどの場合、表現が大きくなりすぎ効率が悪い。従って、能率の良い表現法の考案が重要である。多出力関数を、 $F = (f_0, f_1, \dots, f_{m-1}) : B^n \rightarrow B^m, B = \{0, 1\}$ と定義する。ここで、 n は論理回路の入力数、 m はその出力数を表す。BDD を用いた多出力関数の表現法は、いくつか知られているが [11, 14, 15, 16], 本論文では、ECFN(Encoded Characteristic Function for Non-zero output)[17] を用いる。以下 RBDD は、ECFN を表現する BDD を示し、RMDD と QRMDD は、この RBDD より生成されたものと仮定する。

2.3 ベンチマーク関数

本論文では、表 2.1 に示す 131 個のベンチマーク関数を用いて実験を行なった。表中の n と m は、それぞれ入力数、出力数を表す。また表中の *sequential* 以降のベンチマーク関数は、本来、順序回路を表現する関数である。しかし、本論文では、組合せ回路のみを対象としているので、順序回路の FF(Flip-Flop) を取り除き、それを入出力とした組合せ回路に変換した。本来の関数名に c を付け加え、変換後の組合せ回路の関数名にしている。ECFN の符号化や変数順序最適化は [18] の結果を用いている。表 2.1 の各関数の各実験データは論文のページ数の制限のために割愛する。

3 QRMDD(k) の節点数

3.1 一般の論理関数

任意の論理関数に対して、次の二つの定理が成立する。

定理 3.1 任意の n 入力論理関数は、節点数が高々

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

の *QRBDD* で表現できる。ここで r は、

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

を満足する最大の整数である。

定理 3.2 任意の n 入力論理関数は、節点数が高々

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

表 2.1: 実験に用いたベンチマーク関数

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	tst2	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	s208c	18	9
bcd	26	38	misg	56	23	s298c	17	20
c8	28	18	mish	94	43	s344c	24	26
cc	21	20	misj	35	14	s349c	24	26
chkn	29	7	mlp10	20	20	s382c	24	27
cht	47	36	mux	21	1	s400c	24	27
cm150a	21	1	my_adder	33	17	s420c	34	17
comp	32	3	opa	17	69	s444c	24	27
cordic	23	2	pair	173	137	s510c	25	13
count	35	16	pcl	19	9	s526c	24	27
cps	24	109	pcler8	27	17	s641c	54	43
dalu	75	16	pd	16	40	s713c	54	42
des	256	245	pm1	16	13	s820c	23	24
dk48	15	17	rckl	32	7	s832c	23	24
duke2	22	29	rot	135	107	s838c	66	33
e64	65	65	sct	19	15	s1196c	32	32
ex4	128	28	seq	41	35	s1423c	91	79
example2	85	66	shift	19	16	s5378c	214	228
exep	30	63	signet	39	8	s9234c	247	250

の *QRMDD*(k) で表現できる。ここで s, t は

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

$$t \geq \frac{n}{k} - s$$

を満足する最小の整数である。

定理 3.1, 3.2 の証明は、付録に添付する。

3.2 ベンチマーク関数

表 3.1 は表 2.1 の各ベンチマーク関数を *QRMDD*(k) で表現した場合の k と節点数の関係を示す。表 3.1 の *ave* は、*QRMDD*(1) の節点数を 1.00 としたときの算術平均を表し、*stdv* は、標準偏差を表す。いま、

表 3.1: k と QRMDD(k) の節点数の関係

	k				
	1	2	3	4	5
<i>ave</i>	1.00	0.50	0.33	0.25	0.20
<i>stdv</i>	0.000	0.014	0.007	0.013	0.009

表 3.2: $\eta \geq 0.1$ の関数

<i>Name</i>	回路の種類	<i>Name</i>	回路の種類
<i>C499</i>	誤り訂正回路	<i>my_adder</i>	加算回路
<i>C1908</i>	誤り訂正回路	<i>pcl</i>	制御回路
<i>comp</i>	比較回路	<i>tcon</i>	制御回路
<i>i3</i>	制御回路	<i>vg2</i>	制御回路
<i>in1</i>	制御回路	<i>vtx1</i>	制御回路
<i>mlp10</i>	乗算回路	<i>x1dn</i>	制御回路

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

とおくと、表 2.1 に示す 131 個のベンチマーク関数のうち 119 個の関数において、 $\eta < 0.1$ が成立している。つまり多くの関数で

性質 3.1

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

が成立している。残り 12 個の関数では、 $\eta \geq 0.1$ である。12 個の関数を表 3.2 に示す。また、関数の入出力数に対して、 k が大きいと η が大きくなり性質 3.1 は成立しない。従って、入出力数が小さい関数は本論文の実験に用いていない。

3.3 乱数関数

性質 3.1 が乱数関数に対して成立しているか否かを調べるために、 n 入力乱数関数を生成した。各 n に対して、 2^{n-1} 個の最小項をランダムに生成した。

表 3.3 は、 n 入力乱数関数を QRMDD(k) で表現したときの節点数の平均値 (サンプル数 10) を示している。そのときの偏差は、表にのせていないが、平均値の $\pm 2\%$ 以内に収まっている。表 3.4 は、定理 3.2 により計算した QRMDD(k) の節点数の上界を示す。

表 3.5 は、 n 入力乱数関数の QRMDD(1) に関する実験値と計算値の差の割合

$$\gamma(\%) = \frac{\text{計算値} - \text{実験値}}{\text{計算値}} \times 100$$

表 3.3: 乱数関数を表現する QRMDD(k) の節点数の平均値

n	k				
	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

表 3.4: QRMDD(k) の節点数の上界

n	k				
	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

を示している。表 3.5 から、 r が変化する付近では γ は大きく、それ以外では、小さいことがわかる。表 3.3 より乱数関数に対しては性質 3.1 は成立しないことがわかる。また、一般の関数に対しても性質 3.1 は、成立しない。

4 QRMDD(k) の面積時間複雑度

4.1 QRMDD(k) を表現するためのメモリ量

QRMDD(k) は、 X_1, X_2, \dots, X_u と順に変数を評価するので、次に評価すべき変数をカウンタで求めることができる。そのため QRMDD(k) をメモリに格納する際、各節点には枝が示すアドレスだけを格納すればよく、インデックスを格納する必要は無い。一方、RMDD(k) で

表 3.5: 実験値と計算値の差の割合 γ (乱数関数)

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

表 4.1: k と $QRMDD(k)$ の A の関係

	k				
	1	2	3	4	5
<i>ave</i>	1.00	0.91	1.14	1.65	2.54
<i>stdev</i>	0.000	0.036	0.070	0.114	0.190

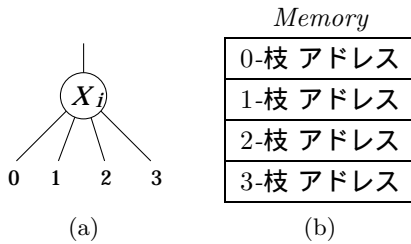


図 4.1: $QRMDD(2)$ の節点のデータ構造

は、次に評価すべき変数はパスによって異なるため、インデックスと各枝のアドレスを格納する必要がある。

例 4.1 図 4.1 に、 $QRMDD(2)$ の節点のデータ構造を示す。また、図 4.2 に、 $RMDD(2)$ の節点のデータ構造を示す。(例終り)

$QRMDD(k)$ の各節点は、 2^k 個の枝を持つので $QRMDD(k)$ の全節点を表現するためには、

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

ワード必要である。一方、 $RMDD(k)$ の全節点を表現するためには、

$$(2^k + 1) nodes(RMDD(k))$$

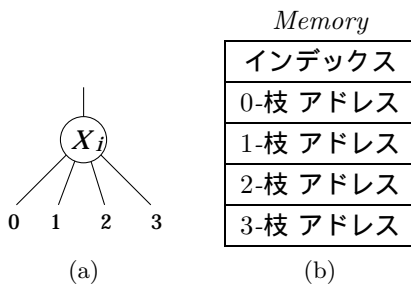


図 4.2: $RMDD(2)$ の節点のデータ構造

ワード必要である。また、 DD をメモリに格納するとき、各節点にアドレスを割り当てる。このアドレスを指定するためのビット数、つまり各ワードのビット数は、

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

必要である。よって $QRMDD(k)$ を表現するための総メモリ量は、

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

と計算できる。一方、 $RMDD(k)$ を表現するための総メモリ量は、

$$(2^k + 1) nodes(RMDD(k)) \lceil \log_2 nodes(RMDD(k)) \rceil$$

で計算できる。

4.2 $QMRDD(k)$ の面積時間複雑度

$QRMDD(k)$ では k 変数を同時に評価できるので、 $QRMDD(k)$ を用いると、 $QRMDD(1)$ に比べメモリアクセス回数を、 k 分の 1 に削減できる。一方、必要なメモリ量は $\frac{2^k}{k}$ に比例して増加する。ここでは、面積時間複雑度という尺度 [4, 19] を用いて $QRMDD(k)$ における最適な k の値を考察する。

定義 4.1 面積時間複雑度 (Area-Time complexity) とは、計算を行うために必要な面積と時間の総合的な尺度であり、

$$AT = (\text{面積}) \times (\text{時間})$$

および、

$$AT^2 = (\text{面積}) \times (\text{時間})^2$$

で定義する。

本論文では、面積として $QRMDD(k)$ を表現するために必要なメモリ量を考え、時間として $QRMDD(k)$ を評価するために必要なメモリアクセス回数を考える。

尺度 AT は、メモリ量とメモリアクセス数の両方が重要視されるときに用いる。一方、 AT^2 は、メモリ量より

表 4.2: k と QRMDD(k) の AT の関係

	k				
	1	2	3	4	5
<i>ave</i>	1.00	0.45	0.38	0.41	0.51
<i>stdev</i>	0.000	0.018	0.023	0.028	0.038

表 4.3: k と QRMDD(k) の AT^2 の関係

	k						
	1	2	3	4	5	6	7
<i>ave</i>	1.000	0.227	0.127	0.103	0.101	0.116	0.141
<i>stdev</i>	0.000	0.009	0.008	0.007	0.008	0.013	0.016

もメモリアクセス数のほうが重要視されるときに用いる。例えば AT は、組み込みシステム [2] を構成する際の尺度、 AT^2 は論理シミュレータ [6, 7] を構成する際の尺度として利用できる。

4.3 実験結果

表 2.1 のベンチマーク関数に対して、QRMDD(k) を表現するための総メモリ量 A 、尺度 AT 、そして尺度 AT^2 を調べた。表 4.1 に、 k と A の関係を、表 4.2 に、 k と AT の関係を、そして表 4.3 に、 k と AT^2 の関係を示す。表の見方は、表 3.1 と同様である。

表 2.1 の 131 個のベンチマーク関数において、総メモリ量 A は、 $k = 2$ のとき最小で、尺度 AT は、 $k = 3$ または $k = 4$ のとき最小となった。また、尺度 AT^2 は、 $k = 4 \sim 6$ のときに最小となった。

4.4 解析

4.3 節では、QRMDD(k) において、 A 、 AT および AT^2 の値を最小にする k の値を実験的に求めた。ここでは、性質 3.1 が成立すると仮定して、解析的に求めてみよう。面積 A を QRMDD(k) を表現するために必要な総メモリ量、時間 T を QRMDD(k) を評価するために必要なメモリアクセス回数とすると

$$A = 2^k \text{nodes}(\text{QRMDD}(k)) \lceil \log_2 \text{nodes}(\text{QRMDD}(k)) \rceil$$

$$T = \left\lceil \frac{n}{k} \right\rceil$$

となる。ここで、性質 3.1 が成立すると仮定し、 $\text{nodes}(\text{QRMDD}(1)) = N$ とおくと、

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

を得る。同様に、

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

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

が成立する。 N は、QRBDD の節点数で、数 100 以上であり、一方 $k = 1 \sim 7$ 程度なので、近似式

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

を成立する。これより、 A 、 AT と AT^2 の式は、

$$A \simeq \frac{2^k}{k} C_0,$$

$$AT \simeq \frac{2^k}{k^2} C_1,$$

$$AT^2 \simeq \frac{2^k}{k^3} C_2$$

と簡単化できる。ここで、 C_0 、 C_1 、 C_2 は k の値に依存しない数である。このとき、 A 、 AT および AT^2 の値を最小にする k の値は、それぞれ $k = 2$ 、 $k = 3$ および $k = 4$ である。

5 結論とコメント

本論文では、QRMDD(k) (Quasi-Reduced Multi-valued Decision Diagram using k bits) を用いた論理関数の表現法に関して以下のことを考察した。1) QRMDD(k) の節点数の上界を求める公式を導いた。また、乱数関数を多数生成し、QRMDD(k) で表現したときの節点数を求め、公式との差について考察した。2) 多くのベンチマーク関数において、性質 3.1 が成立することを示した。3) QRMDD(k) の総メモリ量、および評価時間は k の値に依存するため、使用目的に応じた最適な k の値を求めることが必要である。本論文では、実験と考察により $k = 3 \sim 6$ のとき面積時間複雑度が最小になることを示した。

謝辞

本研究は一部、日本学術振興会、科学研究費補助金による。

参考文献

- [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 multivalued functions with an application to genetic algorithms,” *Proc. of International Symposium on Multiple Valued Logic*, pp. 40-45, May 1994.
- [4] 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.
- [5] R. E. Bryant, “Graph-based algorithms for boolean function manipulation,” *IEEE Trans. Comput.*, Vol. C-35, No. 8, pp. 677–691, Aug. 1986.
- [6] 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)*, Jan. 26-28, Yokohama, Japan.
- [7] Y. Iguchi, T. Sasao, M. Matsuura, “Implementation of multiple-output functions using PQMDDs,” *International Symposium on Multiple-Valued Logic*, pp.199-205, May 2000.
- [8] 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.
- [9] 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.
- [10] 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.
- [11] 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.
- [12] D. M. Miller, “Multiple-valued logic design tools,” *Proc. of International Symposium on Multiple Valued Logic*, pp. 2–11, May 1993.
- [13] T. Sasao, “FPGA design by generalized functional decomposition,” (*Sasao ed.*) *Logic Synthesis and Optimization*, Kluwer Academic Publishers, 1993.
- [14] T. Sasao and M. Fujita (ed.), *Representations of Discrete Functions*, Kluwer Academic Publishers 1996.

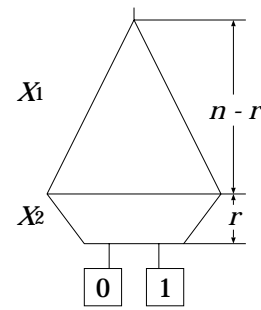


図 A.1: BDD の分割

- [15] T. Sasao and J. T. Butler, “A method to represent multiple-output switching functions by using multi-valued decision diagrams,” *IEEE International Symposium on Multiple-Valued Logic*, pp. 248-254, Santiago de Compostela, Spain, May 29-31, 1996.
- [16] T. Sasao, *Switching Theory for Logic Synthesis*, Kluwer Academic Publishers, 1999.
- [17] T. Sasao, “Compact SOP representations for multiple-output functions: An encoding method using multiple-valued logic,” 31st International Symposium on Multiple-Valued Logic, Warsaw, Poland, May 22 - 24, 2001, pp207-211.
- [18] 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 .
- [19] C. D. Thompson, “Area-Time complexity for VLSI,” *Ann. Symp. on Theory of Computing*, May 1979.

A 付録

A.1 定理 3.1の証明

定義 A.1 n 入力論理関数を表現する *QRBDD* (*Quasi-Reduced ordered Binary Decision Diagram*) を図 A.1のように上下に分割する. 分割した上の部分を上段部とよび, 下の部分を下段部とよぶ. また, 上段部は, $n - r$ 個の変数 $X_1 = (x_1, x_2, \dots, x_{n-r})$ をもち, 下段部は r 個の変数 $X_2 = (x_{n-r+1}, \dots, x_n)$ をもつとする.

定理 3.1の証明 n 入力論理関数を表現する完全二分木の非終端節点数は, $2^n - 1$ である. 下段部では同じ関数を表現する節点は共有するため, 幅が狭くなる. r が $2^{n-r} \geq 2^r$ を満たす最大の整数のとき, BDD の節点数は最小となる [9]. ここで下段部を考えると, 最大 2^{2^r} 個の r 入力論理関数を表現可能である. これらの全ての関

数を表現したとき下段部の節点数は最大となる。このとき BDD は,

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

の形で論理関数を表現している。ここで $X_1 = (x_1, x_2, \dots, x_{n-r})$ かつ, $X_2 = (x_{n-r+1}, x_{n-r+2}, \dots, x_n)$ である。また,

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

である。上段部で $X_1^{\vec{a}_i}$ を実現し, 下段部で $f(\vec{a}_i, X_2)$ を実現する。 $f(\vec{a}_i, X_2)$ は, r 変数関数なので高々 2^{2^r} 種類あればよい。下段部では, 終端節点から上に r 個の変数までの各 i において, 2^{2^i} 個の関数を実現すれば十分である。従って QRBD 節点数は

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

個あれば十分である。 (証明終り)

A.2 定理 3.2 の証明

定理 3.2 を証明するために, 次の補題を利用する。

補題 A.1 n 入力論理関数を $f(X)$, $X = (x_1, x_2, \dots, x_n)$ とする。 $f(X) = f(X_1, X_2, \dots, X_u)$ を,

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

と分解したときの列複雑度を μ_i , $i = 1, 2, \dots, u$ とする。ただし, $u = \lceil \frac{n}{k} \rceil$ である。 f を実現する QRMDD(k) の節点数は

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

となる。

証明 [13, 14] より QRMDD(k) の変数 X_i における幅は, 列複雑度 μ_i に等しいことがわかる。このことと, 定義 2.5 より補題の成立は明らかである。 (証明終り)

定理 3.2 の証明 n 入力論理関数を表現する QRMDD(k) の節点数は, 補題 A.1 より, 繰り返し関数分解したときの列複雑度の総和となる。

QRBD の上段部では, s 段に分解される。このときの列複雑度 $\mu_1, \mu_2, \dots, \mu_s$ は, 各々 $1, 2^k, 2^{2k}, \dots, 2^{s k}$ となり, これは, 初項 1, 公比 2^k の等比数列である。

QRBD の下段部では, t 段に分解される。下段部でも, 節点数は各列複雑度, すなわち QRBD における, 対応する変数の幅の総和により求められる。 $n \neq (s+t)k$ のとき, 変数 X_u は, k 入力とならない。そのため, u 段目は別に計算する。2 値論理関数なので $\mu_u = 2$ となる。

QRMDD(k) の節点数は,

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

個あれば十分である。 (証明終り)