F-050

ベイジアンネットワークを表現するゼロサプレス型BDDの変数順序付けに関する実験と考察
Experiments and Considerations on ZBDD Variable Ordering for Representing Bayesian Networks

礒松 紘平 † Kouhei Isomatsu
湊 真一 † Shin-ichi Minato

1. はじめに

ベイジアンネットワークは，グラフ構造による確率モデルの表現方法の一種であり，近年，様々な用途に広く用いられている [7]．与えられたベイジアンネットワークと観測データに対して，ネットワーク内の各確率変数の確率分布を計算することは，実用上しばしば現れる問題であり，様々な高速化方法が研究されている [2].

最近，著者らは，VLSI CAD の分野で大規模論理関数データの表現方法として広く用いられている二分決定グラフ（BDD：Binary Decision Diagrams）[1]，その中でも「ゼロサプレス型 BDD（ZBDD：Zero-suppressed BDD）[6] と呼ばれるデータ構造を用いて、ベイジアンネットワークを表現し，効率よく確率計算を行う手法を提案した [5]. 本稿では，ベイジアンネットワークを表現する ZBDD の変数順序づけに関する実験と考察を行った.

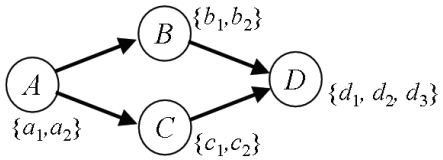
これまで、ZBDD における変数の順序付けが，生成される ZBDD の大きさに多大な影響を与えることがわかっているが，ベイジアンネットワークを表現する場合の順序付けによる影響に関してはあまり研究がされていなかった。本実験では，ベイジアンネットワークから生成された ZBDD に対して変数順序改善を行うことで ZBDD のサイズにどのような変化が見られるのかについて，実験と考察を行う.

2. ベイジアンネットワークと MLF 式

ベイジアンネットワーク（以後 BN と書く）は図 1 に示すような確率モデルを表現する非巡回有向グラフである．各々のノード（BN ノードと呼ぶ）は，それぞれ独立した個別の確率変数 X を持っている．X は一般に多値の変数であり，$\{x_1, x_2, ...x_k\}$ のいずれかの値を取るものとする．また各 BN ノードは，上流側の BN ノードの確率変数の値に依存する条件付き確率テーブル（CPT と呼ばれる）を持ち，これにより確率変数の確率分布が表現されている.

BN の確率分布を計算するための方法の 1 つとして，Multi-Linear Function（MLF）と呼ばれる数式を生成する方法が知られている。MLF は，1 つの確率変数 x に対して，2 種類の論理変数を使って表現する．1 つは X の値を表現する λ 変数と，X の確率分布の数値を記号的に表現する θ 変数である．以下に MLF の例を示す.

$$
\begin{aligned}
& \lambda_{a_1}\lambda_{b_1}\lambda_{c_1}\lambda_{d_1}\theta_{a_1}\theta_{b_1|a_1}\theta_{c_1|a_1}\theta_{d_1|b_1c_1} \\
+\ & \lambda_{a_1}\lambda_{b_1}\lambda_{c_1}\lambda_{d_2}\theta_{a_1}\theta_{b_1|a_1}\theta_{c_1|a_1}\theta_{d_2|b_1c_1} \\
+\ & \lambda_{a_1}\lambda_{b_1}\lambda_{c_1}\lambda_{d_3}\theta_{a_1}\theta_{b_1|a_1}\theta_{c_1|a_1}\theta_{d_3|b_1c_1} \\
+\ & \lambda_{a_1}\lambda_{b_1}\lambda_{c_2}\lambda_{d_1}\theta_{a_1}\theta_{b_1|a_1}\theta_{c_2|a_1}\theta_{d_3|b_1c_2} \\
+\ & \ldots \\
+\ & \lambda_{a_2}\lambda_{b_2}\lambda_{c_2}\lambda_{d_3}\theta_{a_2}\theta_{b_2|a_2}\theta_{c_2|a_2}\theta_{d_3|b_2c_2}.
\end{aligned}
$$



A	$\mathrm{Prb}(A)$
a_1	$\theta_{a_1} = 0.4$
a_2	$\theta_{a_2} = 0.6$

AB	$\mathrm{Prb}(B\|A)$
a_1b_1	$\theta_{b_1\|a_1} = 0.2$
a_1b_2	$\theta_{b_2\|a_1} = 0.8$
a_2b_1	$\theta_{b_1\|a_2} = 0.8$
a_2b_2	$\theta_{b_2\|a_2} = 0.2$

AC	$\mathrm{Prb}(C\|A)$
a_1c_1	$\theta_{c_1\|a_1} = 0.5$
a_1c_2	$\theta_{c_2\|a_1} = 0.5$
a_2c_1	$\theta_{c_1\|a_2} = 0.5$
a_2c_2	$\theta_{c_2\|a_2} = 0.5$

BCD	$\mathrm{Prb}(D\|B,C)$
$b_1c_1d_1$	$\theta_{d_1\|b_1c_1} = 0.0$
$b_1c_1d_2$	$\theta_{d_2\|b_1c_1} = 0.5$
$b_1c_1d_3$	$\theta_{d_3\|b_1c_1} = 0.5$
$b_1c_2d_1$	$\theta_{d_1\|b_1c_2} = 0.2$
$b_1c_2d_2$	$\theta_{d_2\|b_1c_2} = 0.3$
$b_1c_2d_3$	$\theta_{d_3\|b_1c_2} = 0.5$
$b_2c_1d_1$	$\theta_{d_1\|b_2c_1} = 0.0$
$b_2c_1d_2$	$\theta_{d_2\|b_2c_1} = 0.0$
$b_2c_1d_3$	$\theta_{d_3\|b_2c_1} = 1.0$
$b_2c_2d_1$	$\theta_{d_1\|b_2c_2} = 0.2$
$b_2c_2d_2$	$\theta_{d_2\|b_2c_2} = 0.3$
$b_2c_2d_3$	$\theta_{d_3\|b_2c_2} = 0.5$

図 1: ベイジアンネットワークの例

与えられた BN の MLF 式を生成すれば，ある観測データに対する確率変数の確率分布を自動的に求めることができる．すなわち λ 変数のうち，観測データと矛盾する物に 0 を代入し，それ以外は 1 を代入すると，残った θ 変数の数式が，その事象が起きる確率を計算する算術式になっている．この確率計算に要する時間は，MLF 式の長さに比例するが，一般に MLF 式は元の BN サイズに対して指数関数的に大きくなるため，その計算は容易ではない．ただし，MLF 式をうまく因数分解して，コンパクトな算術式として表現することができれば，確率計算を高速化することができる.

3. ZBDD による BN 表現

3.1 BDD

BDD は，図 2 に示すような論理関数のグラフによる表現である．一般に，論理関数のそれぞれの変数について，0, 1 の値を代入した結果を，二分岐の枝（0-枝/1-枝）で場合分けし得られる論理関数の値を、2 値の定数節点（0-終端節点/1-終端節点）で表現すると，図 2 のような二分木状のグラフになる．このとき，場合分けする変数の順序を固定し，冗長な節点の削除と等価な節点の共有という 2 つの縮約規則を可能な限り適用することにより,「規約」な形が得られ，論理関数をコンパクト且つ一意に表せることが知られている．複数の論理関数を表す BDD の間においても，変数順序を固定すればグラフを共有することが可能である.

BDD は，多くの実用的な論理関数を比較的少ない記

†北海道大学大学院情報科学研究科

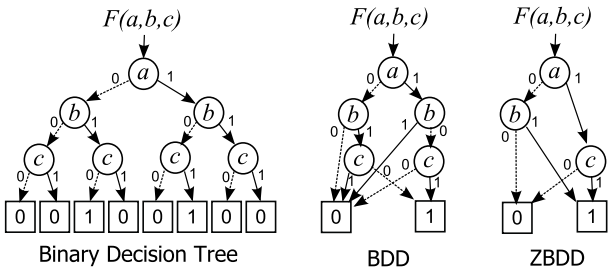


図 2: 二分決定木と BDD，ZBDD

憶量で一意に表現することができる．また，2 つの BDD を入力とし，それらの二項論理演算の結果を表す BDD を直接生成するアルゴリズム [1] が考案されている．このアルゴリズムはハッシュテーブルを巧みに用いることで，データが計算機の主記憶に収まる限りは，その記憶量にほぼ比例する時間内で論理演算を効率よく実行できる．

3.2 ZBDD

BDD は元々は論理関数を表現するために考案されたものだが，これを用いて組み合わせ集合データを表現・操作することも出来る．組合せ集合とは,「n 個のアイテムから任意個を選ぶ組み合わせ」を要素とする集合である．これを BDD で表現するとき、類似する組合せが多ければ，部分的に共通する組合せがグラフ上で共有されて，記憶量や計算時間が大幅に削減される場合がある．さらに，組合せ集合に特化した「ゼロサプレス型 BDD（ZBDD）」[6] を用いると，より簡潔な表現が得られ，一層効率よく扱うことが出来る．

ZBDD では，冗長な節点を削除する簡約化規則が通常の BDD と異なり、1-枝が 0-終端節点を直接指している節点を取り除く，という規則になっている．これにより ZBDD では図 2 のように，組合せ集合に一度も選ばれることのないアイテムに関する節点が自動的に削除されることになり，BDD よりも効率よく組合せ集合を表現・操作することが出来る．

3.3 ZBDD による BN 表現

MLF 式は λ 変数と θ 変数の値からなる多項式で，それぞれの項が単に変数の組合せであるため，組合せ集合と見なすことが出来る．よって ZBDD としてコンパクトに表現することが出来，また，確率計算においても ZBDD のサイズに比例した時間で行うことが出来る．例えば，図 1 のノード B についての MLF 式を見てみると，以下のようになる．

$$\begin{aligned}\mathrm{MLF}(B) &= \lambda_{a_1}\lambda_{b_1}\theta_{a_1}\theta_{b_1|a_1} + \lambda_{a_1}\lambda_{b_2}\theta_{a_1}\theta_{b_2|a_1} \\ &+ \lambda_{a_2}\lambda_{b_1}\theta_{a_2}\theta_{b_1|a_2} + \lambda_{a_2}\lambda_{b_2}\theta_{a_2}\theta_{b_2|a_2}.\end{aligned}$$

ここで，等しいパラメーターが同じ変数を共有するようにパラメーター変数を置き換える．

$$\begin{aligned}\mathrm{MLF}(B) &= \lambda_{a_1}\lambda_{b_1}\theta_{a(0.4)}\theta_{b(0.2)} + \lambda_{a_1}\lambda_{b_2}\theta_{a(0.4)}\theta_{b(0.8)} \\ &+ \lambda_{a_2}\lambda_{b_1}\theta_{a(0.6)}\theta_{b(0.8)} + \lambda_{a_2}\lambda_{b_2}\theta_{a(0.6)}\theta_{b(0.2)}.\end{aligned}$$

MLF(B) による ZBDD の一例を図 3 に示す．この例ではルートノードから 1 の終端ノードまで 4 通りの行き方があり，それぞれの変数が MLF 式の項に相当している．これは MLF 式の暗黙の表現である．ZBDD の構造は MLF 式を更に因数分解したコンパクトな形となっている．なお，MLF 式は一つの確率変数が多くの λ 変数，θ 変数を持つため，図 3 の様に，関係のある変数を並べて配置している．ZBDD のノードは，図 4 が示すように，それぞれ単純な変換規則を備えた算術演算手順と解釈することが出来る．これは，MLF 式の ZBDD を生成した後に、ZBDD のサイズに比例する回数の算術演算手順が容易に得られるという事を意味している．文献 [5] に書かれている BN の代表的なベンチマーク例題を ZBDD で表現した際の表を次に示す．（http://www.cs.huji.ac.il/labs/compbio/Repository）

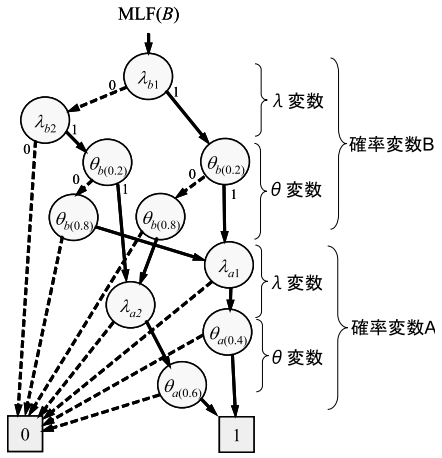


図 3: MLF(B) の ZBDD 表現の例

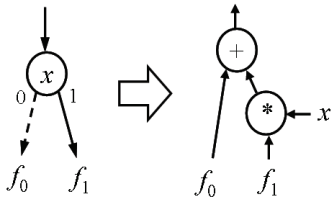


図 4: ZBDD から算術演算手順への簡約化

4. ZBDD の変数順序改善法の適用

BN が中規模以上になると、ZBDD を用いても MLF 式を表現することが困難になる．そこで，ZBDD をコンパクトに表現するための変数順序付が重要である．本実験ではよい順序付けを求めるための手法として以前用いた「変数順序改善法」を用いて実験と考察を行った．ZBDD の性質として，変数の順序がサイズに大きく影響していることが分かっている．変数順序改善法とはこの順序をより良いものにして ZBDD のサイズを小さくするという手法である．以前頻出パタン集合を現す ZBDD に関してこの手法を用いて ZBDD のノード数の改善を試みた．この方法に類するものとして，過去に隣接変数の交換に基く逐次改善手法 [3] やアニーリング法 [4] 等の手法が提案されている．具体的な方法として，ある変数の順序で ZBDD を作成し，そこから変数の入れ替えを

表 1: ベンチマーク例題の ZBDD サイズ [5]

BN name	BN nodes	indicator vars.	parameter vars.	offline compile |ZBDDs| (total)
alarm	37	105	187	34,299
hailfinder	56	223	835	294,605
mildew	35	616	6,709	15,310,511
pathfinder(pf1)	109	448	1,839	16,808
pathfinder(pf23)	135	520	2,304	17,557
pigs	441	1,323	1,474	73,543
water	32	116	3,578	25,629
munin1	189	995	4,249	–
munin2	1,003	5,376	22,866	9,936,191
munin3	1,044	5,604	24,116	11,191,778
munin4	1,041	5,648	24,242	5,724,468

表 2: BN データに対する変数順序改善の効果

BN name	BN データのサイズの変化		
	改善前	改善後	交換回数
alarm	19213	19038	6
hailfinder	108450	107847	22
pathfinder(pf1)	106	106	0
pathfinder(pf23)	979	975	3
pigs	8863	8801	4
water	13396	13396	0

行い，ZBDD のノード数を減らす，ということを行う．本実験ではこの変数順序改善法を BN データに適用し，BN データのサイズがどのように変化するかを調べた．また，ZBDD 作成する際の変数の初期順序は BN データでの変数の出現順となっており深さ優先順となっている．

4.1 変数の交換アルゴリズム

本実験では ZBDD で表現した幾つかの BN データに対して，隣り合う 2 変数の交換を行い，サイズがどのように変化するか調べた．具体的には，n 個の変数について，i 番目の変数と，$i+1$ 番目の変数を交換する．交換前より ZBDD のサイズが小さくなればその交換を採用する．その交換を $i = 1$〜$n-1$ までの変数について繰り返し行い，1 回のループの中でサイズが小さくなる交換が見つからなくなれば終了するという手法である．

5. 実験結果と考察

本実験において使用した PC は Celeron(R)，2.8GHz，SuSE Linux10.3，主記憶 1536Mbyte で ZBDD の最大ノード数は 1000 万個とした．
ZBDD の変数順序改善法を用いた BN データの変数順序改善のための実験として，代表的な BN データについて，初期順序を深さ優先としたデータベース出現順として，変数順序改善法を用いて，ZBDD のノード数がどれだけ変化するのかを調べた．その結果を表 2 に示す．
ここで初期ノード数は交換を行う前のノード数，終了ノード数は交換により変化したノード数，交換回数は実際に交換が行われた回数である．

この結果を見ると，まず交換回数が極めて少ない事が分かる．これは隣り合う変数同士の交換による改善が殆ど見られないと言う事である．これは同じ確率変数に属する λ 変数，θ 変数は相関関係が強いので近接させておくのが好ましいと言う事であると考えられる．また変数交換を行う前と後では ZBDD のサイズには大きな変化が見られない．多少改善されている部分もあるが，これはそのブロック内での交換の際に変数の順序が改善されたと考えられる．

6. おわりに

本稿では，BN を表現する ZBDD に対して変数順序改善法を用いた場合，どのような結果になるか述べた．MLF 式から生成された ZBDD に関して，変数順序改善法を用いて隣接した変数同士の入れ換えを行ったが大きな縮約効果は得られなかった．この事から BN データに関して，同じ確率変数に属する λ 変数と θ 変数のブロックを無視した交換では，あまり高い効果が得られないと言う事が分かった．またブロック内の順序に関しても，効果は限定的なものであるということが分かった．以上の事からブロックごとの交換を行う事が必要であると感じた．但し，ブロックごとにただ交換するだけではその交換の過程で ZBDD のサイズが大きくなり，ZBDD の生成自体が困難になる事が予想されるので，入れ換えを行う際にも何らかの処置が必要であると思われる．今後はそれらの処置の方法も含めて，ブロック単位での変数交換を行い，BN データのサイズがどの様に変化するか実験する予定である．

謝辞

ご討論いただいた Thomas Zeugmann 教授に感謝いたします．本研究の一部は，日本学術振興会科研費 萌芽研究「二分決定グラフに基づく大規模ベイジアンネットワーク解析処理法の研究」（代表：湊真一，課題番号 20650017）による．

参考文献

[1] R.E. Bryant, Graph-based algorithms for Boolean function manipulation, IEEE Trans. Comput., C-35, 8(1986), 677-691.

[2] M. Chavira and A. Darwiche,"Compiling Bayesian Networks with Local Structure," In Proc.19th International Joint Conference on Artificial Intelligence(IJCAI-2005),pp.1306-1312,Aug.2005.

[3] M. Fujita, Y. Matsunaga, and T. Kakuda, On variable ordering of binary descision diagrams for the application of multi-level logic synthesis, In Proc. IEEE European Design Automation Conf. (EDAC-91), (1991), 50-54.

[4] N. Ishiura, H. Sawada, and S. Yajima, Minimization of binary descision deagrams based on exchange of variables, In Proc. ACM/IEEE International Conf. on Computer-Aided Design(ICCAD-91), (1991), 472-475.

[5] S. Minato, K. Satoh, and T. Sato, "Compiling Bayesian Networks by Symbolic Probability Caluculation Based on Zero-suppressed BDDs" In Proc. 19th nternational Joint Conference on Artificial Intelligence(IJCAI-2005),pp.2550-2555,Aug.2005.

[6] S. Minato, Zero-suppressed BDDs for set manipulation in combinatorial prpblems, In Proc.30th ACM/IEEE Design Automation Conf.(DAC-93), (1993), 272-277.

[7] Richard E. Neapolitan, "Learning Bayesian Networks", Pearson Education, Inc.,2004.