



HOKKAIDO UNIVERSITY

Title	Propositionalizing the EM algorithm by BDDs
Author(s)	Ishihata, Masakazu; Kameya, Yoshitaka; Sato, Taisuke et al.
Citation	Transactions of the Japanese Society for Artificial Intelligence, 25(3), 475-484 https://doi.org/10.1527/tjsai.25.475
Issue Date	2010
Doc URL	https://hdl.handle.net/2115/47337
Type	journal article
File Information	18_25_475.pdf



BDD 上の命題化計算に基づく EM アルゴリズム

Propositionalizing the EM algorithm by BDDs

石島 正和
Masakazu Ishihata

東京工業大学大学院情報理工学研究科
Graduate School of Information Science and Engineering, Tokyo Institute of Technology
ishihata@mi.cs.titech.ac.jp, <http://mi.cs.titech.ac.jp/ishihata/>

亀谷 由隆
Yoshitaka Kameya

(同 上)
kameya@mi.cs.titech.ac.jp, <http://mi.cs.titech.ac.jp/kameya/>

佐藤 泰介
Taisuke Sato

(同 上)
sato@mi.cs.titech.ac.jp, <http://sato-www.cs.titech.ac.jp/sato/>

湊 真一
Shin-ich Minato

北海道大学大学院情報科学研究科
Graduate School of Information Science and Technology, Hokkaido University
minato@ist.hokudai.ac.jp, <http://www-alg.ist.hokudai.ac.jp/~minato/>

keywords: machine learning, EM algorithm, binary decision diagram (BDD), propositionalized probability computation

Summary

We propose an Expectation-Maximization (EM) algorithm which works on binary decision diagrams (BDDs). The proposed algorithm, *BDD-EM algorithm*, opens a way to apply BDDs to statistical learning. The BDD-EM algorithm makes it possible to learn probabilities in statistical models described by Boolean formulas, and the time complexity is proportional to the size of BDDs representing them. We apply the BDD-EM algorithm to prediction of intermittent errors in logic circuits and demonstrate that it can identify error gates in a 3bit adder circuit.

1. はじめに

近年、自然言語や遺伝子配列、関係データベースなどの記号データ中の不確実性をどのように扱うかが問題となっている。これを扱う 1つの方法としてベイジアンネットワーク (BN) や隠れマルコフモデル (HMM), 確率的文脈自由文法 (PCFG) などの記号データを扱う確率モデルを構築し、そのモデルに基づき確率推論を行う方法がある。BN, HMM, PCFG はそれぞれ異なる確率推論/学習アルゴリズムが提案されており、BN は Belief Propagation, HMM は Forward-Backward アルゴリズム, PCFG は Inside-Outside アルゴリズムが用いられている。一方、命題論理式を用いて確率モデルの確率計算や学習を行う命題化計算 [佐藤 07] によれば BN, HMM, PCFG を命題化することにより、各モデルにおける確率推論を 1つのアルゴリズムにより実現できる。更に、命題論理を用いることで BN では表現しにくい Context-Specific Independence [Boutilier 96] やゼロ確率といった情報を効率的に表現できることが分かっている。このように命題化計算は BN, HMM, PCFG を含む多様な確率モデルを効率的に表現できる。しかし、この命題論理式で表現される確率モデルに対する学習アルゴリズムはまだ発展途上であり、命題化計算に基づく確率モデリング言語である PRISM [Sato 97] は計算効率のため扱える論理式にいくつかの制約を設けている。

本論文では任意の命題論理式で表現される確率モデルに対する学習アルゴリズムとして *Binary Decision Diagram* (BDD) [Akers 78, Bryant 86] 上で動作する *Expectation-Maximization* (EM) アルゴリズム [Dempster 77] である **BDD-EM アルゴリズム** を提案する。BDD は論理関数のコンパクトな表現として広く知られ、EM アルゴリズムは不完全データに対する最尤推定を行うアルゴリズムとして統計的機械学習で広く利用されている。更に本論文ではより効率的な確率計算を実現するために *decomposed BDD* を導入する。VLSI/CAD 設計の分野では論理関数 F を表現する BDD を構築する際に、 F 中のいくつかの部分関数を *decomposed point* と呼ばれる新しい変数で置き換え、予め部分関数を表現する BDD を個別に構築し、最終的にそれらを統合することで F の BDD を効率よく構築する方法が提案されている [Jain 96]。上記の手法の途中で現れる個別の BDD の集合を *decomposed BDD* と呼ぶ。我々は *decomposed BDD* を統合することなく、個々の BDD をサブルーチンのように呼び出すことによって効率的な確率計算を実現することを提案する。

BDD-EM アルゴリズムは一般的かつ効率的なアルゴリズムである。ここで一般的というのは BDD-EM アルゴリズムは任意の命題論理式に対して適用可能であることを意味し、効率的というのはその計算量は命題論理式を表現する *decomposed BDD* の総節点数に比例することを意味する。

本論文の構成は以下の通りである。まず、2 章では我々が扱う問題設定について述べ、3 章で EM アルゴリズムを定式化する。そして、4 章で BDD を簡単に説明し、5 章で BDD 上で動作する EM アルゴリズムを提案する。6 章では提案手法を論理回路中の故障素子の位置を推測する問題に適用し、提案手法の動作を確認する。最後に関連研究とまとめについて 7 章、8 章で述べる。

2. 問題設定

本章では本論文が扱う問題を定式化する。我々は確率的事象の生起を事象に対応する確率的命題変数 V を用いてそれぞれ $V=1$ (起きた), $V=0$ (起こらなかった) と表現する。すべての確率的命題変数の集合を $\mathcal{V}$ と書き、 $\mathcal{V}$ は互いに排他的な 2 つの集合 $\mathcal{X}$, $\mathcal{Y}$ に分割できるとする ($\mathcal{V}=\mathcal{X}\cup\mathcal{Y}$, $\mathcal{X}\cap\mathcal{Y}=\phi$)。 $\mathcal{X}$, $\mathcal{Y}$ の定義はそれぞれ以下の通りである。

$\mathcal{X}=\{X_1, \dots, X_{|\mathcal{X}|}\}$ に含まれる命題を**基本命題**と呼び、各 X_i は互いに独立して値 $x_i \in \{0, 1\}$ を取る。ここで基本命題の中には互いに独立ではあるが同一の確率分布に従う変数も存在する。例えば 2 つの基本命題 $X_i, X_j \in \mathcal{X}$ が同一の分布に従うとき、 X_i と X_j は i.i.d (独立同分布) であるという。基本命題集合 $\mathcal{X}$ を互いに i.i.d. である部分集合 $\pi \subseteq \mathcal{X}$ の集まりに分割したとき、個々の部分集合 π を **i.i.d. 集合**と呼ぶ。つまり、同一の i.i.d. 集合 π に属するすべての基本命題 $X_i, X_j \in \pi$ は互いに i.i.d. である。そしてすべての i.i.d. 集合の族を Π と表現する。 $X \in \mathcal{X}$ が属する i.i.d. 集合を π_X と表し、 X が従う確率分布を **パラメータ** $\theta_{(\pi_X, x)}$ ($x \in \{0, 1\}$, $0 \leq \theta_{(\pi_X, x)} \leq 1$) を用いて以下で表現する。

$$P(X=x)=\theta_{(\pi_X, x)} \quad (\theta_{(\pi_X, 0)} + \theta_{(\pi_X, 1)} = 1)$$

一方、 $\mathcal{Y}=\{Y_1, \dots, Y_{|\mathcal{Y}|}\}$ に含まれる命題を**複合命題**と呼び、各 Y_i に対して以下の論理式が常に成り立つとする。

$$Y_i \Leftrightarrow \text{Body}_i(\mathcal{X}^i \cup \mathcal{Y}^i) \quad \mathcal{X}^i \subseteq \mathcal{X}, \mathcal{Y}^i \subseteq \{Y_j | Y_j \in \mathcal{Y}, j < i\}$$

ここで上式の右辺 $\text{Body}_i(\mathcal{X}^i \cup \mathcal{Y}^i)$ は $\mathcal{X}^i \cup \mathcal{Y}^i$ 中の変数から成る論理式であり、複合命題 Y_i の値は $\text{Body}_i(\mathcal{X}^i \cup \mathcal{Y}^i)$ によって関数的に決定される。本論文では簡単のため $\text{Body}_i(\mathcal{X}^i \cup \mathcal{Y}^i)$ を Y_i の**論理式**、 $\text{Body}_i(\mathcal{X}^i \cup \mathcal{Y}^i)$ が表現する論理関数を Y_i の**論理関数**と呼ぶ。 $Y_i \in \mathcal{Y}$ の論理関数は階層的に定義されており、 $\mathcal{X}$ の値の割り当て $\phi=\{(X_i, x_i)\}_{i=1}^{|\mathcal{X}|}$ ($X_i \in \mathcal{X}$, $x_i \in \{0, 1\}$) が決定すると $Y_1, \dots, Y_{|\mathcal{Y}|}$ はそれぞれの論理関数から値が順次決定し、その時の Y_i の値を $Y_i(\phi)$ と書く。ただし $Y_i, Y_j \in \mathcal{Y}$ ($i < j$) の論理式が共通の基本命題を含む場合、それらは同一の分布に従う独立な変数 (i.i.d) であるとし、独立として扱えない場合は Y_k ($k > i$) の論理式中に現れるすべての Y_i に対して Y_i の論理式を代入することで Y_i を削除し、 Y_i を $\mathcal{Y}$ から取り除く。

我々は複合命題の部分集合 $\mathcal{F} (\subseteq \mathcal{Y})$ 中の命題のみ観測可能であるとし、 $\mathcal{F}$ に含まれる命題変数を観測命題と呼ぶ。本論文では観測データとして系列データ $\mathcal{O}=\{(F^{(t)}, f^{(t)})\}_{t=1}^T$ ($F^{(t)} \in \mathcal{F}, f \in \{0, 1\}$) が与えられる。ただし、観測系列の各要素は互いに独立であると考え、この時、我々の目的は観測系列 $\mathcal{O}$ を $\mathcal{O}$ を得る確率を最大化するように $X \in \mathcal{X}$ が値 $x \in \{0, 1\}$ を取る確率 $\theta_{(\pi_X, x)}$ を $\mathcal{O}$ から学習することである。

この問題設定において複合命題 Y の論理式は、確率的事象 Y がどのようにして生起するかを表した知識に対応する。つまり、事前にその生起に関する知識があるような問題ではそれを命題論理式で表現する。例として次のような遅刻に関する問題を考える。学生 A, B はときどき学校に遅刻し、 A, B の共通の友人 C は彼らが遅刻したかどうかを観測する。ただし、彼らは毎朝同じ授業を履修しているとは限らないので、 C は A, B 両方の遅刻の有無を観測できる日もあれば、どちらか一方しか観測できない日もある。例えば C の 5 日間の観測が「 A は遅刻した」「 B は遅刻した」「 A は遅刻しなかった」「 A, B ともに遅刻した」「 B は遅刻しなかった」であったとする。すると命題 L_A, L_B, L_{AB} をそれぞれ「 A は遅刻した」「 B は遅刻した」「 A, B ともに遅刻した」と定義すれば、観測列 $\mathcal{O}$ は $\{(L_A, 1), (L_B, 1), (L_A, 0), (L_{AB}, 1), (L_B, 0)\}$ と表現できる。次に A, B が遅刻する原因を考えよう。 A は晴れの日には自転車、雨の日にはバスで通学し、 B は毎日バスを利用しているとする。道路が渋滞するとバスが遅れ、バスに乗っている者は遅刻する。また、 A, B はともに寝坊したときは遅刻する。そして 2 人の寝坊の原因は彼らの時計が壊れている場合か彼らが疲れていた場合の 2 つに限るとする。このように例題では A と B の遅刻の生起に関する知識は既に知られており、これら知識は以下のように命題論理式で表現できる。

$$L_{AB} \Leftrightarrow L_A \wedge L_B$$

$$L_A \Leftrightarrow O_A \vee (R \wedge H)$$

$$L_B \Leftrightarrow O_B \vee H$$

$$O_A \Leftrightarrow C_A \vee T_A$$

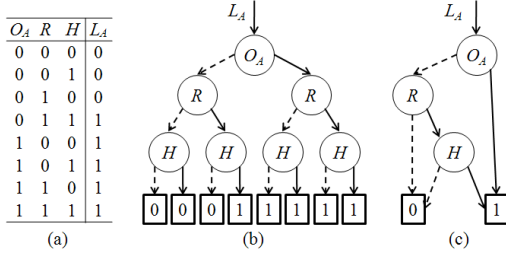
$$O_B \Leftrightarrow C_B \vee T_B$$

ここで H, R, O_Z, C_Z, T_Z ($Z \in \{A, B\}$) はそれぞれ「道路が渋滞する」「雨が降る」「 Z が寝坊する」「 Z の時計が壊れている」「 Z が疲れている」を表す確率的命題変数である。そして問題における基本命題集合 $\mathcal{X}$ 、複合命題集合 $\mathcal{Y}$ 、観測命題集合 $\mathcal{F}$ はそれぞれ以下のようになる。

$$\mathcal{X} = \{H, R, T_A, T_B, C_A, C_B\}$$

$$\mathcal{Y} = \{O_A, O_B, L_A, L_B, L_{AB}\}$$

$$\mathcal{F} = \{L_{AB}, L_A, L_B\}$$

図 1 L_A を表す (a) 真理値表, (b)BDT, (c)ROBDD

この例における我々の目標は、観測系列 $\mathcal{O}$ を得る確率を最大化する基本命題 H, R, C_Z, T_Z ($Z \in \{A, B\}$) が真偽を取る確率を学習することである。ここで我々は観測命題の値のみを観測可能であり、基本命題集合 $\mathcal{X}$ に対する値の割り当てを観測することはできない。本論文ではこのような不完全な観測データから最尤法を用いてパラメータを学習するため、EM アルゴリズム [Dempster 77] を用いる。EM アルゴリズムの詳細については 3 章で説明する。

また、本論文では効率的な確率学習を実現するために論理関数をコンパクトに表現する有向非循環グラフである *Binary Decision Diagram* (BDD) [Akers 78, Bryant 86] を用いる。例えば遅刻の例題の L_A に注目すると L_A の値は O_A, R, H の 3 つの確率的命題変数の値から関数的に決定し、その論理関数は $L_A \Leftrightarrow O_A \vee (R \wedge H)$ という論理式で表現される。これらの確率的命題変数の同時分布をベイジアンネットワークで表現すると、 L_A の条件付確率表は L_A の真理値表 (図 1(a)) に対応する。真理値表のサイズは変数の数に対して指数サイズになるため効率的な確率学習が行えない。しかし、BDD を用いると L_A の論理関数は図 1 (c) のようにコンパクトに表現できる。このように知識が論理形式で与えられる場合は、それをベイジアンネットワークで書き下すよりも命題論理式のまま表現し、論理をコンパクトに扱える構造を用いて学習を行う方が効率的であり、人間にとっても記述/解釈しやすいといえる。本論文では BDD を用いることで効率的な確率学習を実現する。BDD の詳細については 4 章で説明する。

3. EM アルゴリズム

2 章で定義した問題では、各時刻について観測した観測命題とその値の対の列 $\mathcal{O} = \{(F^{(t)}, f^{(t)})\}_{t=1}^T$ が与えられる。しかし、そのときの基本命題集合 $\mathcal{X}$ に対する値の割り当て ϕ を知ることはできない。このような不完全な観測データから最尤法を用いて確率値を学習するアルゴリズムとして EM アルゴリズム [Dempster 77] が知られている。EM アルゴリズムは観測列 $\mathcal{O}$ の対数尤度を最大化する各基本命題 $X \in \mathcal{X}$ が値 $x \in \{0, 1\}$ をとる確率 $\theta_{(\pi_X, x)}$ を *Expectation step* (E-step) と *Maximization step*

(M-step) を繰り返すことにより推定する。ここでは 2 章で定式化された問題に対する EM アルゴリズムを示す。

まず、EM アルゴリズムの最大化の対象である観測列 $\mathcal{O}$ の対数尤度の計算式を導出する。観測命題 $F \in \mathcal{F}$ は基本命題集合 $\mathcal{X}$ に対する値の割り当て ϕ に対して関数的に値が決まるため、 $X \in \mathcal{X}$ が真偽をとる確率 $\theta_{(\pi_X, x)}$ ($x \in \{0, 1\}$) から観測列 $\mathcal{O}$ を得る確率を計算できる。すべてのパラメータの集合を $\theta = \{\theta_{(\pi, x)} \mid \pi \in \Pi, x \in \{0, 1\}\}$ と書くと、 $\mathcal{X}$ が割り当て ϕ が指定した値をとる確率 $P_\theta(\phi)$ は以下のように計算される。

$$P_\theta(\phi) = \prod_{(X, x) \in \phi} \theta_{(\pi_X, x)}$$

更に、 Φ を $\mathcal{X}$ に対するすべての値の割り当ての集合、 $\Phi_{F=f}$ ($F \in \mathcal{F}, f \in \{0, 1\}$) を $\phi \in \Phi$ のうち $F(\phi) = f$ となる割り当ての集合とすれば、観測列 $\mathcal{O}$ を得る確率 $P_\theta(\mathcal{O})$ は以下のように θ を用いて計算できる。

$$\begin{aligned} P_\theta(\mathcal{O}) &= \prod_{(F^{(t)}, f^{(t)}) \in \mathcal{O}} P_\theta(F^{(t)} = f^{(t)}) \\ &= \prod_{(F^{(t)}, f^{(t)}) \in \mathcal{O}} \sum_{\phi \in \Phi_{F^{(t)}=f^{(t)}}} P_\theta(\phi) \\ &= \prod_{(F^{(t)}, f^{(t)}) \in \mathcal{O}} \sum_{\phi \in \Phi_{F^{(t)}=f^{(t)}}} \prod_{(X, x) \in \phi} \theta_{(\pi_X, x)} \end{aligned}$$

ここで観測対 (F, f) が観測列 $\mathcal{O}$ に表れる回数を $\delta_{\mathcal{O}}(F, f)$ と表すと、現在のパラメータ θ における観測列 $\mathcal{O}$ に対する対数尤度 $L_{\mathcal{O}}(\theta)$ は次式で与えられる。

$$\begin{aligned} L_{\mathcal{O}}(\theta) &= \log P_\theta(\mathcal{O}) \\ &= \sum_{F \in \mathcal{F}, f \in \{0, 1\}} \delta_{\mathcal{O}}(F, f) \log \left[\sum_{\phi \in \Phi_{F=f}} \prod_{(X, x) \in \phi} \theta_{(\pi_X, x)} \right] \quad (1) \end{aligned}$$

EM アルゴリズムの一般論により、観測列 $\mathcal{O}$ の対数尤度 $L_{\mathcal{O}}(\theta)$ を極大化するパラメータ $\hat{\theta}$ は以下の E-step と M-step を $L_{\mathcal{O}}(\theta)$ が収束するまで交互に繰り返すことにより推定される。

E-step: 以下で定義される条件付き期待値 $\eta_{\theta}^x[\pi]$ ($\pi \in \Pi, x \in \{0, 1\}$) を計算する。

$$\eta_{\theta}^x[\pi] = \sum_{F \in \mathcal{F}, f \in \{0, 1\}} \delta_{\mathcal{O}}(F, f) \sum_{\phi \in \Phi_{F=f}} \sigma_{\pi, x}(\phi) P_\theta(\phi | F = f) \quad (2)$$

ここで $\sigma_{\pi, x}(\phi) = |\{X \mid X \in \pi, (X, x) \in \phi\}|$ である。

M-step: パラメータ $\theta_{(\pi, x)}$ ($\pi \in \Pi, x \in \{0, 1\}$) を

$$\hat{\theta}_{(\pi, x)} := \eta_{\theta}^x[\pi] / (\eta_{\theta}^1[\pi] + \eta_{\theta}^0[\pi]) \quad (3)$$

に更新する。

式 (2) の条件付き期待値 $\eta_{\theta}^x[\pi]$ は「観測列 $\mathcal{O}$ を得たとき、 $X \in \pi$ が値 x で現れる回数の期待値」である。

2 章の遅刻の例題に対して EM アルゴリズムを実行しよう。まず基本命題 $\mathcal{X} = \{H, R, C_A, C_B, T_A, T_B\}$ を i.i.d. 集合に分割する。 C_A と C_B は A, B の時計が壊れていることを表しているが、 A と B は同じ型の時計を使っていると仮定すれば、 C_A と C_B は i.i.d. であると考えることができる。すると $\mathcal{X}$ は $\pi_1 = \{H\}$, $\pi_2 = \{R\}$, $\pi_3 = \{C_A, C_B\}$, $\pi_4 = \{T_A\}$, $\pi_5 = \{T_B\}$ の 5 つの i.i.d. 集合に分割できて、パラメータは $\theta_{(\pi_i, x)}$ ($1 \leq i \leq 5$, $x \in \{0, 1\}$) の 10 個となる。その結果この例題の E-step は 10 個の条件付き期待値 $\eta_{\theta}^x[\pi_i]$ を計算し、M-step は $\eta_{\theta}^x[\pi_i]$ を用いて 10 個のパラメータ $\theta_{(\pi_i, x)}$ を更新する。この 2 ステップを式 (1) で計算される対数尤度が十分に収束するまで交互に繰り返すことで、観測列 $\mathcal{O}$ の対数尤度 $L_{\mathcal{O}}(\theta)$ を極大化するパラメータ $\hat{\theta}$ が得られる。

しかし、ここで定式化した EM アルゴリズムを実行するには計算上の困難がある。対数尤度と条件付き期待値の定義式 (1)(2) 中の $\Phi_{F=f}$ ($F \in \mathcal{F}$, $f \in \{0, 1\}$) は F の真理値表の $F=f$ となる行の集合に対応する。このため式 (1)(2) の計算量は基本命題の数 k に対して指数オーダーとなる。本論文ではこの困難を解決するため、論理関数をよりコンパクトに表現できる BDD を用いて確率、条件付き期待値計算を効率的に行う方法を提案する。4 章で BDD について説明し、5 章で提案手法について説明する。

4. Binary decision diagram

4.1 Binary decision diagram

Binary decision diagram (BDD) [Akers 78, Bryant 86] は論理関数をコンパクトに表現する有向非循環グラフである。BDD 中のすべての節点 n にはラベルがついており、そのラベルを $Var(n)$ で表す。ラベル $Var(n)$ が命題変数である節点 n を**変数節点**、 $Var(n)$ が 1 又は 0 である n を**定数節点**と呼ぶ。すべての変数節点 n は **1-枝** と **0-枝** と呼ばれる 2 本の出力枝を持ち、それぞれを $\langle n, 1 \rangle$ と $\langle n, 0 \rangle$ で表す。また、 $\langle n, x \rangle$ ($x \in \{0, 1\}$) が至る子節点を x -child と呼び、 $Ch_x(n)$ と書く。これに対して定数節点は出力枝を持たない。一方始点が存在しない枝を**入力枝**と呼び、入力枝が指す節点を BDD の**根節点**と呼ぶ。BDD 中の根節点から定数節点に至るパスは命題変数の割り当てとその時の論理関数の値を表現している。すべてのパスにおいて現れる命題変数の順序が共通であるとき、その BDD を *ordered BDD* (OBDD) と呼び、その順序を OBDD の**変数順序**と言う。本論文で OBDD のみを扱う。

例として 2 章の例題で与えられた論理式 $L_A \Leftrightarrow O_A \vee (R \wedge H)$ を用いる。図 1(a) と (b) はそれぞれ L_A の論理関数を表現する真理値表と変数順序を $\{O_A, R, H\}$ としたときの OBDD である。図中の実線と破線はそれぞれ 1-枝と 0-枝を表す。図 1(b) のように完全二分木である OBDD のことを**場合分け二分木**と呼ぶ。 L_A の場合分け二分木

上のパス $\{O_A, \langle O_A, 0 \rangle, R, \langle R, 0 \rangle, H, \langle H, 0 \rangle, \boxed{0}\}$ は値の割り当てが $\{(O_A, 0), (R, 0), (H, 0)\}$ であるとき $L_A = 0$ であることを表しており、これは L_A の真理値表の 1 行目に対応する。場合分け二分木上の根節点から定数節点に至るパスは真理値表の各行と 1 対 1 に対応する。

一般に同一の論理関数を表現する OBDD は複数種類存在する。例えば図 1(c) も図 1(b) と同様に L_A の論理関数を表現する OBDD である。図 1(c) の OBDD は (b) の場合分け二分木と比較して節点数が少ないが、これは (b) 中のいくつかの節点を以下の 2 つの**簡略化規則**に従って削除したためである。

削除規則: 節点 n において、 $Ch_1(n) = Ch_0(n)$ のとき n を削除する。

共有規則: 2 つの節点 n, n' において、 $Var(n) = Var(n')$, $Ch_x(n) = Ch_x(n')$ ($x \in \{0, 1\}$) のとき n, n' を共有化する。

それ以上簡略化規則を適用できない OBDD のことを *reduced OBDD* (ROBDD) と呼ぶ。ROBDD は論理関数と変数順序に対して一意に決まる。ROBDD を真理値表に対応する場合分け二分木に対して簡約化規則を適用する方法で構築するとその構築計算量は常に指数的な計算時間がかかってしまう。そこで実用的には論理式から ROBDD を直接構築するアルゴリズムが用いられている [Akers 78, Bryant 86]。

例えば図 1(c) はこれ以上簡略化規則を適用できないため L_A の ROBDD である。図 1(c) 上のパス $\{O_A, \langle O_A, 1 \rangle, \boxed{1}\}$ は、 $O_A = 1$ のとき $L_A = 1$ であることを表すが、これは L_A の真理値表の下 4 行に対応する。このように ROBDD は真理値表を圧縮する効果がある。ROBDD のサイズは変数順序に依存するが、最適な変数順序を見つけることは NP-hard である [Tani 96]。しかしその一方で [Minato 90] などでは良い変数順序を発見するヒューリスティックが提案されている。本論文では特に断りのない限り ROBDD を扱うことにし、以後 ROBDD を BDD と呼ぶ。

4.2 Decomposed BDD

論理関数 f の BDD を効率的に構築する方法として、 f 中の M 個の部分関数を M 個の新しい変数 $g_1, \dots, g_M$ に置き換え、 f と $g_1, \dots, g_M$ を表す BDD を個別に構築し、それらを統合することで f を表現する単一の BDD を得る方法が提案されている [Jain 96]。部分関数 $g_1, \dots, g_M$ を *decomposition point*, 統合する前の BDD の集合を *decomposed BDD* と呼ぶ。

本論文では各複合命題を *decomposition point* として個別の BDD で表現し、それらをサブルーチンの様呼び出すことで効率的な確率計算を実現する。 $Y \in \mathcal{Y}$ の論理関数を表現する BDD を Δ_Y と書き、BDD の系列 $\Delta_{\mathcal{F}} = \{\Delta_{Y_1}, \dots, \Delta_{Y_{|\mathcal{Y}|}}\}$ を $\mathcal{F}$ を表現する *decomposed BDD* と呼ぶ。 Δ_Y の根節点を $Root(Y)$, Δ_Y の変数節点と変数ラベルの集合をそれぞれ $\mathcal{N}_Y = \{n_1, \dots, n_{|\mathcal{N}_Y|}\}$,

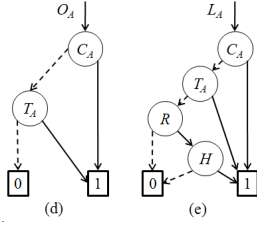


図 2 O_A を表す BDD(d), および図 1 と (d) を統合した BDD

$\mathcal{V}_Y = \{V_1, \dots, V_{|\mathcal{V}_Y|}\}$ と書く. ここで n_i はトポロジカル順序が i 番目の節点, V_i は変数順序が i 番目の変数を意味する. つまり $n_1 = \text{Root}(Y)$ である. ここで $\mathcal{V}_{Y_i}, \mathcal{V}_{Y_j}$ ($Y_i, Y_j \in \mathcal{Y}$) が共通の変数を要素に持つとき, [Jain 96] ではそれら共通変数は同一の値をとると考えるのに対し, 本論文では 2 章で述べたように独立に値をとると考える.

例えば図 2(d) は O_A の論理関数を表現する BDD Δ_{O_A} であり, $\mathcal{N}_{O_A} = \{C_A, T_A\}$, $\mathcal{V}_{O_A} = \{C_A, T_A\}$, $\text{Root}(O_A) = C_A$ である. [Jain 96] では Δ_{O_A} を図 2(c) の Δ_{L_A} に統合することで図 2(e) のような L_A の論理関数を表現する単一の BDD を構築するが, 我々はそれらを統合することなく Δ_{O_A} と Δ_{L_A} のまま利用する.

5. 提案手法

5.1 BDD-EM アルゴリズム

3 章で定式化した EM アルゴリズムを実行するには式 (1) で定義される観測列 $\mathcal{O}$ の対数尤度 $L_{\mathcal{O}}(\theta)$ 及び式 (2) で定義される条件付き期待値 $\eta_{\theta}^x[\pi]$ ($x \in \{0, 1\}$, $\pi \in \Pi_F$) を計算する必要がある. しかし, これらの値は観測命題 $F \in \mathcal{F}$ を表現する真理値表に沿って計算されるため, 計算量が基本命題の数 k に対して指数オーダーとなる. そこで本論文では真理値表を圧縮する効果のある BDD を用いて計算量を削減する **BDD-EM アルゴリズム** を提案する. BDD-EM アルゴリズム (Procedure 1) はまず, E-step において $\mathcal{F}$ を表現する decomposed BDD $\Delta_{\mathcal{F}}$ 上の動的計画法に基づき条件付き期待値 $\eta_{\theta}^x[\pi]$ を計算し, M-step において式 (3) に従ってパラメータ θ を更新する. そして条件付き期待値と同様に $\Delta_{\mathcal{F}}$ 上で対数尤度 $L_{\mathcal{O}}(\theta)$ を計算し, この値が十分に収束するまで E-step と M-step を交互に繰り返す. Procedure 1 中の $\text{BACKWARD}(\Delta_{Y_i})$, $\text{FORWARD}(\Delta_{Y_i})$, $\text{INSIDE}(\Delta_{Y_i})$, $\text{OUTSIDE}(\Delta_{Y_i})$ はそれぞれ, 条件付き確率を効率よく計算するために導入される **後ろ向き確率**, **前向き確率**, **内側確率**, **外側確率** を計算する. 以下でそれらの確率値の意味とその計算方法を示し, 条件付き確率 $\eta_{\theta}^x[\pi]$ がどのように計算されるかを説明する.

5.2 後ろ向き確率と前向き確率

$\text{BACKWARD}(\Delta_Y)$ は Δ_Y 中のすべての節点 $n \in \mathcal{N}_Y$ に対して **後ろ向き確率** $\mathcal{B}_{\theta}^x[n]$ ($x \in \{0, 1\}$) を計算する. $\mathcal{B}_{\theta}^1[n]$

Procedure 1 BDD-EM($\mathcal{X}, \mathcal{Y}, \mathcal{F}, \Pi, \Delta_{\mathcal{F}}, \mathcal{O}$)

```

1: for all  $\pi \in \Pi$ ,  $x \in \{0, 1\}$ 
2:   Initialize  $\theta_{(\pi, x)}$ ;
3: end for
4: repeat
5:   % E-step
6:   for  $i = 1$  to  $|\mathcal{Y}|$  do
7:      $\text{BACKWARD}(\Delta_{Y_i})$ ;
8:      $\text{FORWARD}(\Delta_{Y_i})$ ;
9:      $\text{INSIDE}(\Delta_{Y_i})$ ;
10:   end for
11:    $\text{INITOUTSIDE}(\mathcal{Y}, \mathcal{F}, \Pi, \mathcal{O})$ ;
12:   for  $i = |\mathcal{Y}|$  to 1 do
13:      $\text{OUTSIDE}(\Delta_{Y_i})$ ;
14:   end for
15:    $\text{EXPECTATION}(\Pi)$ ;
16:   % M-step
17:   for all  $\pi \in \Pi$ ,  $x \in \{0, 1\}$ 
18:      $\theta_{(\pi, x)} := \eta_{\theta}^x[\pi] / (\eta_{\theta}^1[\pi] + \eta_{\theta}^0[\pi])$ ;
19:   end for
20:    $L_{\mathcal{O}}[\theta] = \text{LOGLIKELIHOOD}(\mathcal{Y}, \mathcal{F}, \Delta_{\mathcal{F}}, \mathcal{O})$ ;
21: until Log-likelihood  $L_{\mathcal{O}}[\theta]$  converges;

```

Procedure 2 $\text{BACKWARD}(\Delta_Y)$

```

1:  $\mathcal{B}_{\theta}^1[\boxed{1}] = 1$ ;  $\mathcal{B}_{\theta}^1[\boxed{0}] = 0$ ;  $\mathcal{B}_{\theta}^0[\boxed{1}] = 0$ ;  $\mathcal{B}_{\theta}^0[\boxed{0}] = 1$ ;
2: for  $i = |\mathcal{N}_Y|$  to 1 do
3:    $V = \text{Var}(n_i)$ ;
4:   if  $V \in \mathcal{X}$  then
5:      $\mathcal{B}_{\theta}^f[n_i] = \sum_{x \in \{0, 1\}} \theta_{(\pi_V, x)} \mathcal{B}_{\theta}^f[\text{Ch}_x(n_i)]$ ; ( $f \in \{0, 1\}$ )
6:   else if  $V \in \mathcal{Y}$  then
7:      $\mathcal{B}_{\theta}^f[n_i] = \sum_{x \in \{0, 1\}} \mathcal{P}_{\theta}^x[V] \mathcal{B}_{\theta}^f[\text{Ch}_x(n_i)]$ ; ( $f \in \{0, 1\}$ )
8:   end if
9: end for

```

($\mathcal{B}_{\theta}^0[n]$) は n から 1 定数節点 (0 定数節点) に至る確率を表す. 例えば図 1(c) の BDD の節点 H を見てみると, H から 1 定数節点に至るパスは $\{H, \langle H, 1 \rangle, \boxed{1}\}$ の一つである. すなわち H の後ろ向き確率 $\mathcal{B}_{\theta}^1[H]$ は $\theta_{(\pi_H, 1)}$ となる. 後ろ向き確率 $\mathcal{B}_{\theta}^x[n]$ は Procedure 2 のように n の子節点 $\text{Ch}_x(n)$ の後ろ向き確率を用いて動的に計算できる. 従って $\text{BACKWARD}(\Delta_Y)$ の計算量は $O(|\mathcal{N}_Y|)$ となる.

一方で $\text{FORWARD}(\Delta_Y)$ は Δ_Y 中のすべての節点 $n \in \mathcal{N}_Y$ に対して **前向き確率** $\mathcal{F}_{\theta}[n]$ を計算する. $\mathcal{F}_{\theta}[n]$ は Δ_Y の根節点 $\text{Root}(Y)$ から n に至る確率を表す. 例えば図 1(c) の BDD の節点 H を見ると, $\text{Root}(Y)$ から H に至るパスは $\{O_A, \langle O_A, 0 \rangle, R, \langle R, 1 \rangle, H\}$ の一つである. よって H の前向き確率 $\mathcal{F}_{\theta}[H]$ は $\theta_{(O_A, 0)} \theta_{(R, 1)}$ となる. 前向き確率 $\mathcal{F}_{\theta}[n]$ も後ろ向き確率と同様に Δ_Y 上で動的に計算でき (Procedure 3), その計算量は $|\mathcal{N}_Y|$ に比例する.

5.3 内側確率と外側確率

内側確率 $\mathcal{P}_{\theta}^y[Y]$ ($y \in \{0, 1\}$, $Y \in \mathcal{Y}$) は複合命題 Y が値 y をとる確率であり, これは後ろ向き確率 $\mathcal{B}_{\theta}^y[\text{Root}(Y)]$

Procedure 3 FORWARD(Δ_Y)

```

1:  $\mathcal{F}_\theta[n_1] = 1$ ;
2: for  $i = 2$  to  $|\mathcal{N}_Y|$  do
3:    $\mathcal{F}_\theta[n_i] = 0$ ;
4: end for
5: for  $i = 1$  to  $|\mathcal{N}_Y|$  do
6:    $V = \text{Var}(n_i)$ ;
7:   if  $V \in \mathcal{X}$  then
8:      $\mathcal{F}_\theta[\text{Ch}_x(n_i)] += \theta_{(\pi_V, x)} \mathcal{F}_\theta[n_i]; (x \in \{0, 1\})$ 
9:   else if  $V \in \mathcal{Y}$  then
10:     $\mathcal{F}_\theta[\text{Ch}_x(n_i)] += \mathcal{P}_\theta^x[V] \mathcal{F}_\theta[n_i]; (x \in \{0, 1\})$ 
11:   end if
12: end for

```

と一致する。従って Y の内側確率は Procedure 4 のように計算され、更に観測列 $\mathcal{O}$ を得たときの対数尤度 $L_{\mathcal{O}}[\theta]$ は観測命題 $F \in \mathcal{F}$ の内側確率 $\mathcal{P}_\theta^f[F]$ ($f \in \{0, 1\}$) を用いて Procedure 5 のように計算できる。

一方、**外側確率**は条件付き期待値 $\eta_\theta^x[\pi]$ を計算するための一時的な値である。外側確率 $\mathcal{Q}_\theta^x[\bullet]$ を観測命題 $F \in \mathcal{F}$, i.i.d. 集合 $\pi \in \Pi$, 基本命題 $X \in \mathcal{X}$ に対して

$$\begin{aligned} \mathcal{Q}_\theta^f[F] &= \frac{\delta_{\mathcal{O}}(F, f)}{P_\theta(F=f)} \\ \mathcal{Q}_\theta^x[\pi] &= \sum_{X \in \pi} \mathcal{Q}_\theta^x[X] \\ \mathcal{Q}_\theta^x[X] &= \sum_{F \in \mathcal{F}, f \in \{0, 1\}} \mathcal{Q}_\theta^f[F] \sum_{\phi \in \Phi_{F=f}^{X=x}} \prod_{(X', x') \in \phi \setminus (X, x)} \theta_{(\pi_{X'}, x')} \\ \Phi_{F=f}^{X=x} &= \{\phi \mid \phi \in \Phi_{F=f}, (X, x) \in \phi\} \end{aligned}$$

と定義すれば、式 (2) の条件付き期待値 $\eta_\theta^x[\pi]$ は次式のよう $\theta_{(\pi, x)}$ と $\mathcal{Q}_\theta^x[\pi]$ の積として計算できる。

$$\begin{aligned} \eta_\theta^x[\pi] &= \sum_{F \in \mathcal{F}, f \in \{0, 1\}} \delta_{\mathcal{O}}(F, f) \sum_{\phi \in \Phi_{F=f}} \sigma_{\pi, x}(\phi) P_\theta(\phi | F=f) \\ &= \sum_{F \in \mathcal{F}, f \in \{0, 1\}} \frac{\delta_{\mathcal{O}}(F, f)}{P_\theta(F=f)} \sum_{\phi \in \Phi_{F=f}} \sigma_{\pi, x}(\phi) P_\theta(\phi) \\ &= \sum_{F \in \mathcal{F}, f \in \{0, 1\}} \mathcal{Q}_\theta^f[F] \sum_{X \in \pi} \sum_{\phi \in \Phi_{F=f}^{X=x}} \prod_{(X', x') \in \phi} \theta_{(\pi_{X'}, x')} \\ &= \mathcal{Q}_\theta^x[\pi] \theta_{(\pi, x)} \end{aligned}$$

$\mathcal{Q}_\theta^x[X]$ の定義式中の $\sum_{\phi \in \Phi_{F=f}^{X=x}} \prod_{(X', x') \in \phi \setminus (X, x)} \theta_{(\pi_{X'}, x')}$ は、 $F=f$ かつ $X=x$ なるすべての $\mathcal{X}$ に対する値の割り当て $\phi \in \Phi_{F=f}^{X=x}$ の確率の総和から $X=x$ の確率を取り除いたもの、つまり $X=x$ の外側の確率であると言える。これらの外側確率は更に複合命題 $Y \in \mathcal{Y}$ に対する外側確率 $\mathcal{Q}_\theta^y[Y]$ を導入することで Procedure 6, 7 のように $\mathcal{F}$ の decomposed BDD $\Delta_{\mathcal{F}}$ に沿って階層的に計算でき、その計算量は $\Delta_{\mathcal{F}}$ のサイズ $\sum_{Y \in \mathcal{Y}} |\mathcal{N}_Y|$ に比例する。

条件付き期待値 $\eta_\theta^x[\pi]$ ($x \in \{0, 1\}$, $\pi \in \Pi$) は観測列 $\mathcal{O}$ を得たときに $\mathcal{X}$ に対する値の割り当て ϕ 中に $X \in \pi$ なる基本命題 X が値 x で現れる回数の期待値である。外側

Procedure 4 INSIDE(Δ_Y)

```

1:  $\mathcal{P}_\theta^x[Y] = \mathcal{B}_\theta^x[\text{Root}(Y)]; (x \in \{0, 1\})$ 

```

Procedure 5 LOGLIKELIHOOD($\mathcal{Y}, \mathcal{F}, \Delta_{\mathcal{F}}, \mathcal{O}$)

```

1: for  $i = 1$  to  $|\mathcal{Y}|$  do
2:   BACKWARD( $\Delta_{Y_i}$ );
3:   INSIDE( $\Delta_{Y_i}$ );
4: end for
5:  $L_{\mathcal{O}}[\theta] = 0$ ;
6: for all  $F \in \mathcal{F}, f \in \{0, 1\}$ 
7:    $L_{\mathcal{O}}[\theta] += \delta_{\mathcal{O}}(F, f) \log \mathcal{P}_\theta^f[F]$ ;
8: end for
9: return  $L_{\mathcal{O}}[\theta]$ ;

```

確率の定義より $\eta_\theta^x[\pi]$ は Procedure 8 によって計算され、従って BDD-EM アルゴリズムの E-step の計算量は $\mathcal{F}$ を表現する decomposed BDD $\Delta_{\mathcal{F}}$ のサイズに比例する。

6. 実 験

本章では 5 章で提案した BDD-EM アルゴリズムが任意の論理式で表現される確率モデルに適用可能であることを示すため、3bit 加算回路の故障箇所診断問題に対して提案手法を適用し、BDD-EM アルゴリズムが正しく動作することを確認する。

6.1 問題の定式化

はじめに本章で扱う論理回路の故障箇所診断問題を定式化する。論理回路中の素子はある確率で 0 または 1 に縮退するとし、その縮退確率が 0 より大きい素子を故障素子と呼ぶ。我々の目的は論理回路から得た長さ N の観測列 (入出力値のペアの列) から回路に含まれる故障素子の場所を推測することである。このような確率的故障を含む論理回路は論理プログラムによって定式化することができ、我々は [Poole 93] で提案されている論理プログラムを参考にモデル化する。各素子は各時刻において他の素子とは独立に状態 $S \in \{ok, stk0, stk1\}$ をとり、 $S=ok$ ならば素子は正常に動作し、 $S=stk0$ ($stk1$) のときには素子の出力は 0 (1) に固定される。時刻 T において、素子 G の状態が S であることを $state(G, S, T)$, 素子 G の出力 $o(G)$ が値 $V \in \{0, 1\}$ であることを $val(o(G), V, T)$

Procedure 6 INITOUTSIDE($\mathcal{X}, \mathcal{Y}, \mathcal{F}, \Pi, \mathcal{O}$)

```

1: for all  $V \in \mathcal{Y} \cup \Pi, x \in \{0, 1\}$ 
2:    $\mathcal{Q}_\theta^x[V] = 0$ ;
3: end for
4: for all  $F \in \mathcal{F}, f \in \{0, 1\}$ 
5:    $\mathcal{Q}_\theta^f[F] = \delta_{\mathcal{O}}(F, f) / \mathcal{P}_\theta^f[F]$ ;
6: end for

```

Procedure 7 OUTSIDE(Δ_Y)

```

1: for  $i = 1$  to  $|\mathcal{N}_Y|$  do
2:    $V = \text{Var}(n_i)$ ;
3:    $V^x = \text{Var}(Ch_x(n_i)); (x \in \{0, 1\})$ 
4:    $q^x = \sum_{f \in \{0,1\}} \mathcal{Q}_\theta^f[Y] \mathcal{F}_\theta[n_i] \mathcal{B}_\theta^f[Ch_x(n_i)]; (x \in \{0, 1\})$ 
5:   if  $V \in \mathcal{X}$  then
6:      $\mathcal{Q}_\theta^x[\pi_V] += q^x; (x \in \{0, 1\})$ 
7:   else if  $V \in \mathcal{Y}$  then
8:      $\mathcal{Q}_\theta^x[V] += q^x; (x \in \{0, 1\})$ 
9:   end if
10:  % Compute the expectations of the deleted nodes.
11:   $\mathcal{D}_\theta^+[V] += q^1 + q^0$ ;
12:   $\mathcal{D}_\theta^-[V^x] += q^x; (x \in \{0, 1\})$ 
13: end for
14: % Add the expectations of the deleted nodes to  $\mathcal{Q}_\theta^x[X]$ 
15:  $\delta = \mathcal{D}_\theta^+[V_1]$ ;
16: for  $i = 2$  to  $|\mathcal{V}_Y|$  do
17:    $\delta -= \mathcal{D}_\theta^-[V_i]$ ;
18:   if  $V_i \in \mathcal{X}$  then
19:      $\mathcal{Q}_\theta^x[\pi_{V_i}] += \delta \theta_{(\pi_{V_i}, x)}; (x \in \{0, 1\})$ 
20:   else if  $V_i \in \mathcal{Y}$  then
21:      $\mathcal{Q}_\theta^x[V_i] += \delta \mathcal{P}_\theta^x[V_i]; (x \in \{0, 1\})$ 
22:   end if
23:    $\delta += \mathcal{D}_\theta^+[V_i]$ ;
24: end for

```

Procedure 8 EXPECTATION(Π)

```

1: for  $\pi \in \Pi$  do
2:    $\eta_\theta^\pi[\pi] = \mathcal{Q}_\theta^\pi[\pi] \theta_{(\pi, x)}; (x \in \{0, 1\})$ 
3: end for

```

と表現し、それらの関係を以下の様に表現する。

$$\begin{aligned}
val(o(G), V, T) &\leftarrow state(G, ok, T) \wedge normal(G, V, T) \\
val(o(G), 0, T) &\leftarrow state(G, stk0, T) \\
val(o(G), 1, T) &\leftarrow state(G, stk1, T)
\end{aligned}$$

ここで $normal(G, V, T)$ は時刻 T における素子 G の正常出力が V であることを表す述語である。また端子 P, Q が接続されていることを $conn(P, Q)$ と表現し、このときの Q の値は以下のように表現される。

$$val(Q, V, T) \leftarrow conn(P, Q) \wedge val(P, V, T)$$

論理回路は上記の述語 $conn$ を用いて表現される。例えば図3は半加算回路の回路図とその定義である。半加算回路 $ha(1)$ の時刻 T における入力 i_1, i_2 、出力 o_1, o_2 であることを $obs(i_1, i_2, o_1, o_2, T)$ と表現すると、これは以下のように述語 val を用いて表現できる。

$$\begin{aligned}
obs(i_1, i_2, o_1, o_2, T) &\Leftrightarrow \\
&val(i(1, ha(1)), i_1, T) \wedge val(i(2, ha(1)), i_2, T) \\
&\wedge val(o(1, ha(1)), o_1, T) \wedge val(o(2, ha(1)), o_2, T)
\end{aligned}$$

これらの関係式を用いることで論理回路から得られた観測は $state(G, S, T)$ の論理式で表現できる。例えば論理

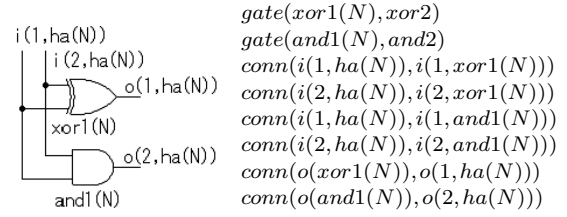


図3 半加算回路とその定義

回路 $ha(1)$ から得た観測 $obs(0, 0, 0, 1, t_1)$ は以下の論理式で表現できる。

$$\begin{aligned}
obs(0, 0, 0, 1, t_1) &\Leftrightarrow \\
&state(xor1(1), ok, t_1) \wedge state(and1(1), stk1, t_1) \\
&\wedge state(xor1(1), stk0, t_1) \wedge state(and1(1), stk1, t_1)
\end{aligned}$$

ただし各素子は状態 $S = \{ok, stk0, stk1\}$ のうち 1 つを取るため、実際の論理式には $state(G, S, T)$ ($S \in \mathcal{S}$) は丁度 1 つが真をとり、残りの 2 つは偽となるよう論理式に制約を加える。

この例では得られた論理式は単純な積和標準形であるが、現実の論理回路は複雑な階層構造となっているため、得られる論理式も複雑な形となる。確率モデリング言語である PRISM[Sato 97] は論理式で表現される確率モデルのパラメータ学習が可能だが、扱う論理式に排反性条件を仮定している。そのため仮定を満たすよう注意深くプログラムを記述する必要がある。本論文では [Poole 93] の論理プログラムを参考にしており、これは PRISM の排反性条件を満たさない。これに対して提案手法は任意の論理式で表現される確率モデルのパラメータ学習が可能であるため、論理プログラムを書き換える必要はない。

上記のように定式化された確率的故障を含む論理回路の故障箇所は 2 章で導入した問題設定に当てはめることで推測できる。論理回路 C の入力、出力数を $\mathcal{I}_C, \mathcal{O}_C$ 、素子の集合を $\mathcal{G}_C$ で表現し、時刻 T に C から得た観測 F_T を $obs(I_1^{(T)}, \dots, I_{\mathcal{I}_C}^{(T)}, O_1^{(T)}, \dots, O_{\mathcal{O}_C}^{(T)}, T)$ と表現すれば、この問題における基本命題 $\mathcal{X}$ 、複合命題 $\mathcal{Y}$ 、観測命題 $\mathcal{F}$ 、観測列 $\mathcal{O}$ はそれぞれ以下のように書ける。

$$\begin{aligned}
\mathcal{X} &= \{state(G, S, T) \mid G \in \mathcal{G}_C, S \in \mathcal{S}, 1 \leq T \leq N\} \\
\mathcal{Y} = \mathcal{F} &= \{F_T \mid 1 \leq T \leq N\} \\
\mathcal{O} &= \{(F_T, 1) \mid 1 \leq T \leq N\}
\end{aligned}$$

各素子の縮退確率は時刻 T に対して不変であると仮定すれば、 $\mathcal{X}$ は以下のような i.i.d. 集合に分割できる。

$$\begin{aligned}
\Pi &= \{\pi_{(G, S)} \mid G \in \mathcal{G}_C, S \in \mathcal{S}\} \\
\pi_{(G, S)} &= \{state(G, S, T) \mid 1 \leq T \leq N\}
\end{aligned}$$

我々はこの問題に対して本論文の提案手法である BDD-EM アルゴリズムを実行することにより $state(G, S, T)$

が真偽をとる確率 $\theta_{(\pi(G,S),x)}$ ($x \in \{0,1\}$) を学習し、縮退確率 $\theta_{(\pi(G,stk0),1)}$, $\theta_{(\pi(G,stk1),1)}$ が 0 より大きいすべての $G \in \mathcal{G}_C$ を故障素子であると推測する。

6.2 実験設定

本節では実験設定を示す。本論文では 3bit 加算回路 C_{3ad} ($\mathcal{I}_{C_{3ad}}=6$, $\mathcal{O}_{C_{3ad}}=4$, $|\mathcal{G}_{C_{3ad}}|=12$) の故障診断を 6.1 節において定式化した論理回路の故障診断問題に当てはめて行う。

本実験では故障素子を最大 3 つ含む 1000 種類の 3bit 加算回路 $C_{3ad}^1, \dots, C_{3ad}^{1000}$ をランダムに生成し、各回路に対して以下の故障箇所推定を行うことで提案手法の推定精度を計算した。ここで故障素子の正常に動作する確率を 0.9, 縮退確率を 0.1 とした。故障確率のうち, 0/1 に縮退する割合は一様乱数で決定した。

- (1) 回路 C_{3ad}^i から長さ N の観測列 $\mathcal{O}^i$ を得る。
- (2) $\mathcal{O}^i$ 中の各観測を表す論理式を得る。
- (3) BDD-EM アルゴリズムを適用する。
- (4) 学習した確率値から C_{3ad}^i の故障箇所を推測する。

なお, EM アルゴリズムの収束判定条件は対数尤度の増加量が 10^{-5} 以下ならば十分に収束しているとした。ここで EM アルゴリズムは局所探索アルゴリズムであるため収束する解が初期値に依存する。そこで本実験では各 $\mathcal{O}^i$ に対して初期値をランダムに変更しながら BDD-EM アルゴリズムを 1000 回実行し、その中で対数尤度が最大となる結果を用いて推測を行った。収束条件を満たすまでに要した平均 EM ステップ数は観測数が 1000 のとき約 40 ステップであった。

観測を表す BDD を構築する際に、変数順序をどのように決めるかは一つの問題となる。論理式や論理回路の構造に基づいた順序づけ法として動的重み付け法 [Minato 90] が知られている。しかし本論文では単純に論理回路の出力に近い素子に関する変数が上位にくるように変数順序を決定した。これは本実験においては動的重み付け法と単純な方法で得られた 2 つの BDD のサイズにあまり差がなかったためである。

6.3 実験結果

本節では 6.2 節で述べた 3bit 全加算回路の故障診断の実験結果を示す。本論文では推測の精度を評価する指標として Recall, Precision, F-measure を用いる。これらの指標は情報検索の分野で検索システムの性能評価に用いられてきたが [Manning 08], 近年では機械学習を用いた予測手法の性能評価などにも利用されている。本実験における Recall は全故障素子のうち故障であると推測されたものの割合を表し, Precision は故障であると推測された素子のうち実際に故障しているものの割合を表す。F-measure は Recall と Precision の調和平均であり, Recall と Precision の双方が高いときに高い値をとる。

図 4 は観測数 N を 10, 50, 100, 500, 1000 と変化させた

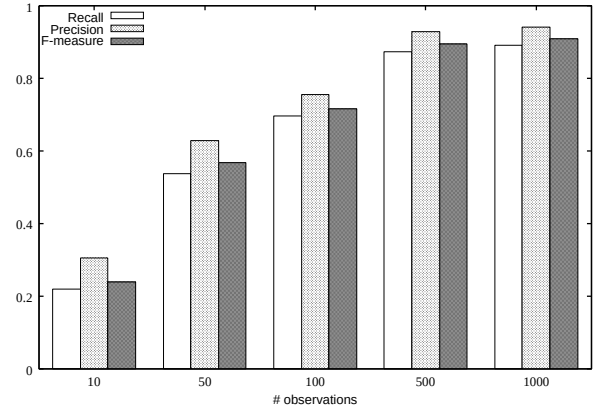


図 4 実験結果

ときの Recall, Precision, F-measure の変化を表したグラフである。図より観測数を増やすに連れて推測の精度が上昇し, $N=1000$ のときに最も高い精度で故障素子を推測できていることが確認できる。この論理回路の故障診断問題では観測を表す論理式は PRSIM が仮定している排反性条件を満たさない。これより提案手法である BDD-EM アルゴリズムは任意の論理式で表現される確率モデルのパラメータ学習が行えることが確認できた。

7. 関連研究

BDD-EM アルゴリズムは命題論理式を用いて確率モデルの確率計算や学習を行う命題化計算 [佐藤 07] の一例であり, 他の例として BN [Minato 07, Chavira 05, Mateescu 05], Markov random field [McAllester 04, Richardson 06], ProbLog [De Raedt 07], PRISM [Sato 97] などが挙げられる。BN の確率計算を BDD の一種である *Zero-suppressed BDD* (ZDD) を用いて行う手法 [Minato 07] が提案されているが, ZDD を用いたパラメータ学習アルゴリズムはまだ提案されていない。本論文では省略したが, BDD-EM アルゴリズムは ZDD 上で動作するように拡張可能である [Ishihata 08]。ProbLog は Prolog プログラムの各述語に確率を導入したプログラミング言語であり, そのプログラムは観測式を説明する基底アトムから成る BDD を用いて確率計算を行う。ProbLog による確率学習アルゴリズムもいくつか提案されているが, [Gutmann 08] は最小二乗法による学習であること, [Thon 08] は EM アルゴリズムによる学習であるが扱える論理関数に制約を設けていることが提案手法と大きく異なる。PRISM [Sato 97] は Prolog を基にした確率モデル記述言語であり, decomposed BDD に似た階層構造を持つ説明グラフというデータ構造を用いることで確率モデルの確率推論や学習を行う。PRISM では計算効率のために扱う論理式の各選言子が互いに排反であるという仮定を置いているが, 本論文で提案した BDD-EM アルゴリズムを用いることでこの仮定を取り除くことが可能である。

上記の研究はいずれも確率モデルに論理表現を導入することで効率的な確率推論，学習を行うことを目的としているが，それとは別に論理的枠組みに確率を導入する試みも存在する．それらの試みの一例として仮説推論によって得られた観測を説明する複数の仮説を確率を用いて評価する研究がいくつかなされている [Poole 93, Inoue 09]．[Poole 93] は複数の仮説を確率を用いて評価する方法を提案しているが，その確率を得る方法については触れていない．本論文では [Poole 93] で挙げられている例題の 1 つである論理回路の故障箇所診断問題に対して BDD-EM アルゴリズムを適用し，各素子が縮退する確率を学習した．[Inoue 09] では薬物の代謝経路に関する複数の仮説を BDD-EM アルゴリズムを用いて評価し，その評価結果が妥当であることを示している．

8. ま と め

本論文は確率的命題変数からなる decomposed BDD 上で EM アルゴリズムを実行する BDD-EM アルゴリズムを提案した．BDD-EM アルゴリズムは汎用の学習アルゴリズムであり，PCFG の Inside-Outside アルゴリズムと HMM の Forward-Backward アルゴリズムのように動的計画法を用いることで効率的に実行される．

更に本論文では BDD-EM アルゴリズムが正しく動作することを確認するため，確率的に振る舞う故障素子を含む論理回路の故障箇所判定問題 [Poole 93] に BDD-EM アルゴリズムを適用した．BDD-EM アルゴリズムは任意の命題論理式に対して適用できることから今回扱った問題に限らず，様々な仮説推論，故障診断によって得られた仮説，診断結果の評価に利用できると期待される．

◇ 付 録 ◇

A. 記 号 一 覧

表 A.1 2 章 問題設定

記号	意味
x, f	0 または 1
$V, \mathcal{V}$	確率的命題変数とその集合
$X, \mathcal{X}$	基本命題とその集合
$Y, \mathcal{Y}$	複合命題とその集合
$F, \mathcal{F}$	観測命題とその集合
π, Π	i.i.d. 集合とその族
π_X	X が属する i.i.d. 集合
$\theta_{(\pi, x)}$	$X \in \pi$ が値 x をとる確率
ϕ	基本命題 $\mathcal{X}$ に対する値の割り当て
$\mathcal{O}$	観測列

表 A.2 3 章 EM アルゴリズム

記号	意味
θ	すべての $\theta_{(\pi, x)}$ の集合
$P_{\theta}(\phi)$	θ の元で $\mathcal{X}$ が値 ϕ を取る確率
Φ	すべての ϕ の集合
$\Phi_{F=f}$	$F(\phi) = f$ を満たす ϕ の集合
$\delta_{\mathcal{O}}(F, f)$	(F, f) が $\mathcal{O}$ 中に現れる回数
$L_{\mathcal{O}}(\theta)$	θ における $\mathcal{O}$ の対数尤度
$\eta_{\theta}^x[\pi]$	条件付き期待値
$\sigma_{\pi, x}(\phi)$	ϕ における $X = x$ ($X \in \pi$) の数

表 A.3 4 章 Binary decision diagram

記号	意味
n	BDD 中の節点
$Var(n)$	n にラベル付けられている変数
$Ch_x(n)$	n の x 子節点
$\langle n, x \rangle$	n の x 枝
Δ_Y	複合命題 Y の論理関数を表現する BDD
$\Delta_{\mathcal{F}}$	$\mathcal{F}$ を表す decomposed BDD
$Root(Y)$	Δ_Y の根節点
$\mathcal{N}_Y$	Δ_Y 中の変数節点集合
$\mathcal{V}_Y$	Y の論理式に現れる変数の集合

表 A.4 5 章 提案手法

記号	意味
$\mathcal{B}_{\theta}^x[n]$	節点 n の後ろ向き確率
$\mathcal{F}_{\theta}[n]$	節点 n の前向き確率
$\mathcal{P}_{\theta}^x[Y]$	複合命題 Y の内側確率
$\mathcal{Q}_{\theta}^x[V]$	変数 V の外側確率
$\mathcal{Q}_{\theta}^x[\pi]$	i.i.d. 集合 π の外側確率

◇ 参 考 文 献 ◇

- [Akers 78] Akers, S. B.: Binary Decision Diagrams, *IEEE Transaction on Computers*, Vol. 27, No. 6, pp. 509–516 (1978)
- [Boutilier 96] Boutilier, C., Friedman, N., Goldszmidt, M., and Koller, D.: Context-Specific Independence in Bayesian Networks, in *Proceedings of 12th Conference on Uncertainty in Artificial Intelligence (UAI'96)*, pp. 115–123 (1996)
- [Bryant 86] Bryant, R.: Graph-based algorithms for Boolean function manipulation, *IEEE Transaction on Computers*, Vol. 35, No. 8, pp. 677–691 (1986)
- [Chavira 05] Chavira, M. and Darwiche, A.: Compiling Bayesian Networks with Local Structure, in *Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI'05)*, pp. 1306–1312 (2005)
- [De Raedt 07] De Raedt, L., Angelika, K., and Toivonen, H.: ProbLog: A Probabilistic Prolog and its Application in Link Discovery, in *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI'07)*, pp. 2468–2473 (2007)
- [Dempster 77] Dempster, A., Laird, N., and Rubin, D.: Maximum likelihood from incomplete data via the EM algorithm, *Journal of the Royal Statistical Society, Series B*, Vol. 39, pp. 1–38 (1977)
- [Gutmann 08] Gutmann, B., Kimmig, A., De Raedt, L., and Kersting, K.: Parameter Learning in Probabilistic Databases: A Least Squares Approach, in *Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML'08)*, Vol. 5211, pp. 473–488 (2008)
- [Inoue 09] Inoue, K., Sato, T., Ishihata, M., Kameya, Y., and Nabeshima, H.: Evaluating abductive hypotheses using an EM algorithm on BDDs, in *Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI'09)*, pp. 810–815 (2009)
- [Ishihata 08] Ishihata, M., Kameya, Y., Sato, T., and Minato, S.: Propositionalizing the EM algorithm by BDDs, *Technical report TR08-0004, Department of Computer Science, Tokyo Institute of Technology* (2008)
- [Jain 96] Jain, J., Narayan, A., Coelho, C., Khatri, S., Sangiovanni-Vincentelli, A., Brayton, R., and Fujita, M.: Decomposition Techniques for Efficient ROBDD Construction, in *Proceedings of the First International Conference on Formal Methods in Computer-Aided Design (FMCAD'96)*, pp. 419–434 (1996)
- [Manning 08] Manning, C. D., Raghavan, P., and Schütze, H.: *Introduction to Information Retrieval*, Cambridge University Press, 1 edition (2008)
- [Mateescu 05] Mateescu, R. and Dechter, R.: The Relationship Between AND/OR Search Spaces and Variable Elimination, in *Proceedings of 21st Conference on Uncertainty in Artificial Intelligence (UAI'05)*, pp. 380–387 (2005)
- [McAllester 04] McAllester, D., Collins, M., and Pereira, F.: Case-Factor Diagrams for Structured Probabilistic Modeling, in *Proceedings of 20th Conference on Uncertainty in Artificial Intelligence (UAI'04)*, pp. 382–391 (2004)
- [Minato 90] Minato, S., Ishiura, N., and Yajima, S.: Shared Binary Decision Diagram with Attributed Edges for Efficient Boolean Function Manipulation, in *Proceedings of the 27th ACM/IEEE Design Automation Conference (DAC'90)*, pp. 52–57 (1990)
- [Minato 07] Minato, S., Satoh, K., and Sato, T.: Compiling Bayesian Networks by Symbolic Probability Calculation Based on Zero-suppressed BDDs, in *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI'07)*, pp. 2550–2555 (2007)
- [Poole 93] Poole, D.: Probabilistic Horn abduction and Bayesian networks, *Artificial Intelligence*, Vol. 64, No. 1, pp. 81–129 (1993)
- [Richardson 06] Richardson, M. and Domingos, P.: Markov logic networks, *Machine Learning*, Vol. 62, No. 1, pp. 107–136 (2006)
- [Sato 97] Sato, T. and Kameya, Y.: PRISM: A symbolic-statistical modeling language, in *Proceedings of the 15th International Joint Conference on Artificial Intelligence (IJCAI'97)*, pp. 1330–1335 (1997)
- [Tani 96] Tani, S., Hamaguchi, K., and Yajima, S.: The Complexity of the Optimal Variable Ordering Problems of a Shared Binary Decision Diagram, *IEICE Transactions on Information and Systems*, Vol. 79, No. 4, pp. 271–281 (1996)

- [Thon 08] Thon, I., Landwehr, N., and De Raedt, L.: A Simple Model for Sequences of Relational State Descriptions, in *Proceedings of the 2008 European Conference on Machine Learning and Knowledge Discovery in Databases (ECML'08)*, pp. 506–521 (2008)
- [佐藤 07] 佐藤泰介, 亀谷由隆: グラフィカルモデルにおける論理的アプローチ, 人工知能学会誌, Vol. 22, No. 3, pp. 306–319 (2007)

[担当委員: 浅井 達哉]

2009 年 11 月 24 日 受理

—— 著 者 紹 介 ——

石 畠 正 和 (学生会員)



2008 年東京工業大学工学部情報理工学研究科卒業, 2010 年同大学大学院情報理工学研究科計算工学専攻修士課程終了, 同年同研究科博士課程進学, 同年より日本学術振興会特別研究員 (DC1). 機械学習, 確率推論の研究に従事. 2008 年度本学会研究会優秀賞受賞, 情報処理学会会員.

亀 谷 由 隆 (正会員)



1995 年東京工業大学工学部情報工学科卒業, 1997 年同大学大学院情報理工学研究科計算工学専攻修士課程修了, 2000 年同研究科博士課程修了, 2001 年新日鉄ソリューションズ株式会社入社, 2003 年より東京工業大学大学院情報理工学研究科計算工学専攻助手 (現在, 助教). 博士 (工学). 機械学習, 確率推論システムの研究に従事.

佐 藤 泰 介 (正会員)



1973 年東京工業大学工学部電子物理学科卒業, 1975 年同大学大学院理工学研究科電気工学専攻修士課程終了, 同年通商産業省工業技術院電子技術総合研究所入所. 1995 年東京工業大学大学院情報理工学研究科に移り, 現在に至る. 工博. 人工知能, 確率モデリングの研究に従事. 日本ソフトウェア科学会会員, 情報処理学会会員, AAAI 会員.

湊 真 一 (正会員)



北海道大学 大学院情報科学研究科 准教授. (独) 科学技術振興機構 ERATO 湊離散構造処理系プロジェクト研究総括 (兼務). 1988 年京都大学工学部情報工学科卒業, 1990 年同大学院修士課程, 1995 年博士課程 (社会人) 修了. 博士 (工学). 1990 年 NTT 入社. 大規模論理データ処理アルゴリズムの研究に従事. 1997 年より 1 年間米国スタンフォード大学客員研究員. 2004 年より現職. BDD (二分決定グラフ) を用いた離散構造の処理に興味を持つ.