決定グラフの自然性について *

小島 健介

2018 年 4 月 12 日

概要

二分決定グラフ (Binary Decision Tree; BDD) とゼロサプレス型二分決定グラフ (Zero-Suppressed BDD; ZDD) の性質の違いを調べる．圏論的な観点からの考察を通して，「BDD はブール関数の表現，ZDD は組合せ集合の表現とみるのが自然である」という主張を数学的に定式化し，証明する．また，BDD の変種である sentential decision diagram についても同様の考察を行う．

1 はじめに

本稿では，**二分決定グラフ** *(Binary Decision Tree; BDD)* [9, 1] と**ゼロサプレス型二分決定グラフ** *(Zero-Suppressed BDD; ZDD)* [10, 14] という二つのデータ構造の違いを考察する．詳しくは 2 節で解説するが，これらはいずれも組合せ集合あるいはブール関数を表現するデータ構造である．組合せ集合とブール関数の間には一対一の対応があるので，その対応のもとで，BDD/ZDD はいずれの表現ともみることができる．しかし，BDD はブール関数の表現とみるのが自然であり，ZDD は組合せ集合の表現とみるのが自然である，とされている．本稿では，このことを数学的な定理として定式化し，証明する．

定式化の概要は次の通りである．まず，ブール関数と組合せ集合の違いを数学的に定式化する．両者の間には一対一の対応があるが，ある種の構造（本稿ではそれを関数の**作用**と呼ぶ）まで考えれば両者は本質的に異なっていることがわかる．この構造を備えている数学的対象は**関手**と呼ばれる．次に，その構造の違いが BDD/ZDD のデータ構造の定義に反映されていることを示す．このことは，関手の間の**自然変換**の概念を用いて記述できる．

また，このような結果は BDD/ZDD に限って成り立つものではなく，それらの拡張である SDD と ZSDD に対しても同様のことが成立する．このことから，本稿で示すブール関数と組合せ集合の区別，およびそれに基づくデータ構造の性質の違いの定式化は，一定の普遍性を備えているものであると予想される．

本稿の構成は以下の通りである．まず 2 節で，BDD と ZDD，およびそれらのブール関数と組合せ集合としての解釈を定義する．続いて 3 節で，関手と自然変換の定義を述べ，BDD/ZDD の解釈が自然変換になっていることを示す．ここまでの議論では，話を簡単にするため，通常の BDD/ZDD で考慮される要素間の順序は考えないが，4 節で要素間の順序を考慮した場合にも類似の結果が成り立つことを説明する．5 節で，SDD/ZSDD に対しても同様の考察を行う．

* 本稿は第 106 回人工知能基本問題研究会の発表資料 [15] に加筆・修正を行ったものです．

2　二分決定グラフ

2.1　組合せ集合とブール関数

集合 X の部分集合のことを X 上の**組合せ** (combination) といい，X 上の組合せの集合のことを，X 上の**組合せ集合** (combination set) という．本稿では X のことを**全体集合**と呼ぶことがある．例えば，全体集合 X として有向グラフ G の辺全体の集合を考えてみよう．G の全域部分木やハミルトン閉路は，G の辺の集合として表現できるから，X 上の組合せとみなせる．したがって，全域部分木全体やハミルトン閉路全体の集合は組合せ集合（と同一視できる集合）の例である．

X 上の組合せ集合全体の集合を $CS(X)$ と書く．混乱のおそれがないときは，組合せをその要素の列で表す．例えば，$\{\{a\},\{b\},\{a,b\}\}$ の代わりに $\{a,b,ab\}$ と書く．また，$\emptyset$ は組合せでも組合せ集合でもあるが，これを組合せとみているときは ε と書いて区別する．

集合 X 上の**ブール関数** (Boolean function) とは，X の要素への真理値の割り当て（X から $\{0,1\}$ への関数）を受け取って 0 または 1 を返す関数のことである．X 上のブール関数全体の集合を $BF(X)$ と書く．ブール関数は X が有限集合であれば論理式で表現できる[*1]．例えば $a \vee b$ は，a と b の少なくとも一方に 1 が割り当てられていれば 1 を，そうでなければ 0 を返すブール関数を表現する．以下，ブール関数と論理式は混乱のおそれがない限り断ることなく同一視する．

組合せ集合とブール関数の間に一対一の対応があることはよく知られている．例えば，上で挙げた $\{a,b,ab\}$ とブール関数 $a \vee b$ は互いに対応している．一般に，次のことが成り立つ．

命題 1. 組合せ集合 $P \in CS(X)$ に対してブール関数 $\tau_X(P) \in BF(X)$ を，各真理値割り当て $\rho\colon X \to \{0,1\}$ に対して

$$\tau_X(P)(\rho) = \begin{cases} 1 & \rho^{-1}(1) \in P \\ 0 & \rho^{-1}(1) \notin P \end{cases}$$

となるものと定義する．このとき，τ_X は $CS(X)$ から $BF(X)$ への全単射である．

上の $\tau_X(P)$ の定義の右辺は，論理式の形では次のように書くことができる．

$$\bigvee_{C \in P} \left(\left(\bigwedge_{x \in C} x \right) \wedge \left(\bigwedge_{x \in X \setminus C} (\neg x) \right) \right)$$

2.2　BDD と ZDD の定義

BDD と ZDD は，組合せ集合（あるいはブール関数）の有向グラフによる表現である．命題 1 により，あるデータを組合せ集合として解釈することとブール関数として解釈することとは同値である（ここでいう同値とは，一方の解釈から他方の解釈が導かれ，この解釈間の変換が互い他の逆になっているという程度の意味である）．

BDD と ZDD は，単なる有向グラフのクラスとしては同じである（これが意味するところを正確に述べると，有向グラフ F に対する二つの述語「F はあるブール関数を表現する BDD である」「F はある組合せ集合

[*1] 以下，断りがない限りは有限集合のみを考えることにするが，X が無限集合の場合でも，本稿の議論は論理式を明示的に用いない形に修正すれば成立する．

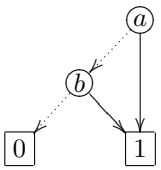

図 1　ダイアグラムの例．丸は分岐節点，四角は終端節点を表し，点線と実線はそれぞれ 0-枝，1-枝を表す．

を表現する ZDD である」は同値である，ということである）．そこで本稿ではまず最初に BDD や ZDD になりうる有向グラフ（**ダイアグラム**と呼ぶことにする）のクラスを定義し，続いてダイアグラムの BDD および ZDD としての解釈を定義する．なお，BDD/ZDD を考えるときには要素間に順序を導入するのが普通だが，それについては 4 節で考察する．

定義 2. 集合 X 上の**ダイアグラム**とは，節点と枝にラベルをもつ閉路のない根付き有向グラフで，以下の条件を満たすものである．

- 節点は**分岐節点** (decision nodes) と**終端節点** (terminal nodes) の二種類に分けられる．
- 分岐節点は X の要素でラベル付けされており，2 本の区別された枝をもつ．これらの枝はそれぞれ 0 と 1 でラベル付けされ，0-**枝** (0-edge)，1-**枝** (1-edge) と呼ばれる．
- 終端節点は枝をもたず，0 または 1 でラベル付けされる（それぞれ 0-**終端節点** (0-terminal node)，1-**終端節点** (1-terminal node) と呼ぶ）．

X 上のダイアグラム全体の集合を $\mathcal{D}(X)$ と書く．ダイアグラムは通常，図 1 のように図示される．

以下，表記を簡単にするために，0-終端節点と 1-終端節点をそれぞれ $\mathbf{0}$ と $\mathbf{1}$ で表す．また (a, F, G) でラベル a をもち 0-枝と 1-枝がそれぞれ F と G を指す分岐節点を表す．例えば，図 1 のダイアグラムをこの記法を用いて書き下すと $(a, (b, \mathbf{0}, \mathbf{1}), \mathbf{1})$ となる．なお，ダイアグラムをこのように表現すると，節点が共有されていたかどうかの情報が失われてしまうが，節点の共有の有無は本稿の議論においては重要ではない．

次に，ダイアグラムを BDD/ZDD として解釈する方法を与える．これはダイアグラムを受け取ってブール関数または組合せ集合を返す関数として定義できる．これらの関数のことを以下では**解釈関数**と呼ぶことにする．

ダイアグラムの BDD としての解釈 β は次のように与えられる．

$$\beta(\mathbf{0}) = \mathit{false} \qquad \beta(\mathbf{1}) = \mathit{true}$$
$$\beta((a, F, G)) = (\neg a \wedge \beta(F)) \vee (a \wedge \beta(G))$$

手続き的に説明すると，関数 $\beta(F)$ は，真理値割り当て ρ が与えられたとき次のように動作する．F の根から探索を始める．探索の各ステップにおいて，今見ている分岐節点のラベル x を取得し，$\rho(x)$ の値が 0 なら 0-枝を，1 なら 1-枝をたどる．この探索処理を終端節点に到達するまで繰り返したのち，その終端節点のラベル（これは 0 または 1 である）を返す．

上の定義の右辺を組合せ集合とみると，終端節点はそれぞれ $\emptyset, \mathcal{P}(X)$ で解釈され，分岐節点 (a, F, G) は

$$\{A \in \beta(F) \mid a \notin A\} \cup \{A \in \beta(G) \mid a \in A\}$$

で解釈されることがわかる．例えば図 1 の BDD としての解釈は，定義から得られる式を同値変形すると $a \vee b$

となることがわかり，これは組合せ集合としては a と b のいずれかが含まれる組合せ全体，すなわち $\{a, b, ab\}$ である．

一方，ZDD としての解釈 ζ は次のように定義される．

$$\zeta(\mathbf{0}) = \emptyset \qquad \zeta(\mathbf{1}) = \{\varepsilon\}$$
$$\zeta((a, F, G)) = \zeta(F) \cup \{A \cup \{a\} \mid A \in \zeta(G)\}$$

ZDD においては，根から 1-終端節点に至る各パスが組合せを表す．各分岐節点において，その節点から 0-枝をたどることは，その節点に付いているラベルを選ばないことを意味し，1-枝をたどることは選ぶことを意味する．パスが表す組合せは，そのパスに沿って要素を選択していったときに最終的に得られる組合せである．そして ZDD が表す組合せ集合は，各パスが表す組合せをすべて集めて得られる集合である．したがって特に，1-終端節点の解釈は $\{\varepsilon\}$，すなわち何も含まない組合せのみからなる組合せ集合となる．また，図 1 を ZDD として解釈すると，$\{a, b\}$ となる．a と b の両方を選択して終端節点に至るパスが存在しないため，BDD の場合と違って ab は含まれない．

2.3 二つの表現の違い

BDD と ZDD の違いは，ダイアグラム中に（より正確にはパス中にだが）現れない要素の扱いにある [10, § 2–3]．それは以下のようにまとめることができる．

X 上の BDD F に一度も現れない要素 $x \in X$ があったとする．このとき，先に述べた $\beta(F)$ の動作に照らして考えると，F が表すブール関数の値の計算に x への割り当ては使われない．したがって，x が真と偽のどちらを割り当てられようと結果には影響しないことになる．これは組合せ集合の言葉に直せば，F が表す組合せ集合にある組合せ C が属するか否かは，$x \in C$ か $x \notin C$ かとは無関係に決まるということである．

一方で ZDD においては，選択することを明示的に指示されたものだけが組合せに入っていると解釈される．したがって，ZDD 中に現れない要素は入らないと解釈されなければならない．つまり ZDD においては BDD とは違い，x が F に現れないときには $x \in C$ であるような組合せ C は $\zeta(F)$ に入らない．

この違いがどのような差を生むかをみるために，例として図 1 のダイアグラムを考え，これを F とする．今までは F を $X = \{a, b\}$ 上のダイアグラムだと暗に仮定していたが，そうではなく $Y = \{a, b, c\}$ 上のダイアグラムと考えることもできる．いずれの場合でも，F を BDD として解釈して得られるブール関数は論理式で書くと $a \vee b$ になるが，これを組合せ集合だと思うと，二つの場合で同じ集合にはならない．実際，X 上で考えれば $\{a, b, ab\}$ だが，Y 上では $\{a, b, ab, ac, bc, abc\}$ となる．また，F を ZDD として解釈すると組合せ集合 $\{a, b\}$ になるが，これをブール関数だと思っても同様のことが起きる．すなわち Y 上で $\{a, b\}$ に対応する論理式は，X 上で考えたときの論理式と $\neg c$ との連言である（c を含む組合せを排除しなければならないため）．よって同じ組合せ集合が X 上と Y 上では異なる論理式で表される．このような事情から「BDD は関数の表現，ZDD は集合の表現」と考えるのが自然であると考えられる．

3 数学的定式化

前節で述べた BDD と ZDD の違いは，ひとことで言い表すと，全体集合 X をより大きい Y に置き換えたときの解釈の変化が BDD と ZDD では異なる，ということだった．このように「考えている集合が変化するとどうなるか」に注目することが定式化の鍵になる．ダイアグラムや組合せ集合，ブール関数という概念を，単に個々の集合 X について定義されているだけでなく，X が変化すればそれに合わせてそれらも何らかのし

かたで変化すると考える．上の例では $X \subseteq Y$ の場合について考えたが，より一般に関数 $f\colon X \to Y$ があると（$X \subseteq Y$ は f が X から Y への包含写像になっている場合である），f に沿ってダイアグラムなどを「変化させる」ことができる．本節では，この状況を圏論 [2] の用語を使って整理する[*2]．

3.1 関数の作用と関手

集合を作る操作 $\mathcal{F}$ であって，上に述べたような「変化」が定まっているものを関手という．正式な定義は以下の通りである．

定義 3. すべての集合 X に対して何らかの集合 $\mathcal{F}(X)$ が与えられていて，かつすべての関数 $f\colon X \to Y$ に対して関数 $\mathcal{F}(f)\colon \mathcal{F}(X) \to \mathcal{F}(Y)$ が与えられているとする．このとき $\mathcal{F}$ が**関手** (functor) であるとは，以下の条件が成り立つことである．

- f が恒等写像ならば $\mathcal{F}(f)$ も恒等写像である．
- すべての関数 f, g について（$f \circ g$ が定義できれば）$\mathcal{F}(f \circ g) = \mathcal{F}(f) \circ \mathcal{F}(g)$ である．

$\mathcal{F}(f)$ のことを，関数 f が集合 $\mathcal{F}(X)$ から $\mathcal{F}(Y)$ への変換を与えているとみて，f の ($\mathcal{F}$ に関する) **作用** (action) と呼ぶことにする [8, § V.7]．なお集合と関数のいずれに対する操作も同じ $\mathcal{F}$ で表すが，どちらを表すかは文脈から判断できるだろう．

例として List という関手を考える．この関手は次のように定義される．

- 集合 X に対して，$\mathsf{List}(X)$ は X の要素の有限列全体からなる集合．
- 関数 $f\colon X \to Y$ に対して，$\mathsf{List}(f)$ は

$$\mathsf{List}(f)([x_1, \ldots, x_n]) = [f(x_1), \ldots, f(x_n)]$$

で定義される関数（Haskell などでいう `map`）．

このとき，List が関手の条件を満たしていることは容易に確かめられる．

3.2 自然変換

関手による関数の作用を保存する変換のことを自然変換という．正式な定義は以下の通りである．

定義 4. $\mathcal{F}, \mathcal{G}$ を関手とする．すべての集合 X に対して関数 $\alpha_X\colon \mathcal{F}(X) \to \mathcal{G}(X)$ が与えられているとき，族 $\alpha = \{\alpha_X\}_X$ を $\mathcal{F}$ から $\mathcal{G}$ への**変換** (transformation) という．さらに，どんな関数 $f\colon X \to Y$ に対しても $\mathcal{G}(f) \circ \alpha_X = \alpha_Y \circ \mathcal{F}(f)$ が成り立つとき，α は**自然変換** (natural transformation) であるという．

f の x への作用を $f \cdot x$ と（$\mathcal{F}(f)$ と $\mathcal{G}(f)$ の区別がつかなくなることに目をつぶって）表すことにすると，自然変換の条件は $\forall x \in \mathcal{F}(X), f \cdot \alpha_X(x) = \alpha_Y(f \cdot x)$ と書ける．このように表現すれば，この条件が作用の保存を表していることが見て取れるだろう．

より直観的には，自然変換とは $\mathcal{F}$ や $\mathcal{G}$ の形のみに依存し，それ以上の余分な情報によらない関数である．例として $\mathcal{F} = \mathcal{G} = \mathsf{List}$ のときを考える．受け取ったリストを逆順にして返す関数は，リストの構造を変更す

[*2] 本節で与える関手と自然変換の定義は，考えている圏がすべて集合と関数の圏 **Sets** の場合のものである．後の節でより一般的な定義を与える．

るだけで，リストの要素が何であるかには依存しない．大雑把にいって，このような関数が自然変換である．一方，リスト中の重複する要素を削除する関数は，リストに何が入っているかによって何番目の要素が削除されるかが異なる．すなわちリストの具体的な要素という「余分な情報」に依存して，返されるリストの構造が変わる．こういう関数は自然変換にはならない．

実は，この「余分な情報によらない関数」であることが，自然変換の定義が本来意図するところである．もともと自然変換は，例えばベクトル空間の間のある線形写像の作り方が基底のとり方に依存するかしないか，というような違いを表すために導入された概念である [6, 7]．基底のとり方に依存しない線形写像は自然であり，依存するものは自然ではない，というような言葉遣いは自然変換という概念が生まれる前から使われていた．基底はベクトル空間の構造（適当な条件を満たす加法とスカラー倍）そのものに内在するわけではなく，同じベクトル空間に対していろいろな基底があるから，その意味で基底は「ベクトル空間一般」を考えている際には余分な情報である．そういう余分な情報を使わないで定義できる対象が「自然である」とされる．そのような「自然さ」の直観とうまく一致しそうな概念として導入されたのが，自然変換である．現在では，上の定義がより直観的な意味での自然さと比較的よく一致するものとして受け入れられている．

3.3 解釈関数の自然さ

次に BDD/ZDD の解釈関数が自然変換になっていることを示す．その前に，ダイアグラムや組合せ集合などの構成を関手に拡張する（すなわち関数の作用を定義する）必要があるが，これは難しくない．

まずダイアグラムの関手 $\mathcal{D}$ を作る．$\mathcal{D}(X)$ はすでに定義した通り，集合 X 上のダイアグラムの集合である．関数 $f\colon X \to Y$ に対して，その作用 $\mathcal{D}(f)$ はラベルの付け替えである（これは List が定める作用が `map` だったことの類似である）．正確には

$$\begin{aligned}&\mathcal{D}(f)(\mathbf{0}) = \mathbf{0}, \quad \mathcal{D}(f)(\mathbf{1}) = \mathbf{1},\\ &\mathcal{D}(f)((a, F, G)) = (f(a), \mathcal{D}(f)(F), \mathcal{D}(f)(G))\end{aligned}$$

と書ける．これが関手の条件を満たすことは容易に確かめられる．

次にブール関数と組合せ集合の関手 BF, CS を定義する．関数 $f : X \to Y$ の作用を

$$\begin{aligned}BF(f)(\varphi(x_1, \ldots, x_n)) &= \varphi(f(x_1), \ldots, f(x_n)),\\ CS(f)(P) &= \{f(A) \mid A \in P\}\end{aligned}$$

と定める．ただし $\varphi(x_1, \ldots, x_n)$ は X 上のブール関数を論理式で表したものであり，$x_1, \ldots, x_n \in X$ である（φ の選び方はいくつもあるが，右辺はブール関数としては φ の選び方によらずに定まる）．これらの定義が関手を定めることも容易に確認できる．

以上の準備のもとで，本稿の主定理は以下のように述べることができる．

定理 5. 1. β は $\mathcal{D}$ から BF への自然変換である．
2. ζ は $\mathcal{D}$ から CS への自然変換である．

Proof. 任意の関数 $f\colon X \to Y$ とダイアグラム $F \in \mathcal{D}(X)$ について $BF(f)(\beta(F)) = \beta(\mathcal{D}(f)(F))$ および $CS(f)(\zeta(F)) = \zeta(\mathcal{D}(f)(F))$ が成り立つことを示せばよい．これらは F の節点数についての帰納法で証明できる． □

ブール関数と組合せ集合が本質的に異なる数学的対象であることの証拠として，次の命題が成り立つ．

命題 6. $BF(X)$ と $CS(X)$ は, 同型だが自然には同型でない．すなわち, 各 X に対して全単射 $\alpha_X\colon BF(X) \to CS(X)$ があるが，この全単射を α が自然変換になるように構成することはできない．

Proof. $X = \{x, y\}$（ただし $x \neq y$）とし，埋め込み $i\colon \emptyset \to X$ と $r(x) = r(y) = x$ で定義される $r\colon X \to X$ を考える．このとき α が自然変換ならば $\alpha_X(true) = \alpha_X(x \vee \neg y)$ である（したがって α_X は単射ではない）ことを示す．

$F = \alpha_\emptyset(true), G = \alpha_X(x \vee \neg y)$ とする．自然性と $CS(i)$ の定義より $F = CS(i)(F) = CS(i)(\alpha_\emptyset(true)) = \alpha_X(BF(i)(true)) = \alpha_X(true)$ となる．よって $F = G$ を示せばよい．また，同様にして $CS(r)(G) = \alpha_X(BF(r)(x \vee \neg y)) = \alpha_X(x \vee \neg x) = \alpha_X(true)$ である．よって $CS(r)(G) = F$ となるから，示すべきことは $CS(r)(G) = G$ と同値である．いま，$F = \alpha_\emptyset(true)$ であることから $F \subseteq \{\emptyset\}$ である．よって既に示した等式から $CS(r)(G) \subseteq \{\emptyset\}$ である．このことから，どんな $C \in G$ に対しても $r(C) \in \{\emptyset\}$ となり，したがって $r(C) = \emptyset$ でなければならない．そのためには $C = \emptyset$ が必要であり，C は任意だったから，$G \subseteq \{\emptyset\}$ がわかる．そのような G に対しては，$CS(r)$ の定義から確かに $CS(r)(G) = G$ が成り立つ． □

この証明は直観的に理解しにくいが，例えば，命題 1 で与えた対応 τ はが自然変換ではないことは容易に確かめられる．X を空でない集合，$i\colon \emptyset \to X$ を包含写像とすると，$BF(i) \circ \tau_\emptyset \neq \tau_X \circ CS(i)$ である（両者は $\{\varepsilon\}$ をそれぞれ $true$ と $\bigwedge_{x\in X}(\neg x)$ に写す）．

なお，$\tau^{-1} \circ \beta$ は $\mathcal{D}$ から CS への変換であるが，これは自然変換にはならない．同様に $\tau \circ \zeta$ も $\mathcal{D}$ から BF への自然ではない変換である．実はより強く，どんな全単射な変換 $\alpha\colon BF \to CS$ を用いても，変換 $\alpha^{-1} \circ \beta$ と $\alpha \circ \zeta$ は自然にはできないことが示せる．このことは，BDD を組合せ集合として解釈する，あるいは ZDD をブール関数として解釈するのが自然ではないことを意味していると考えられる．これを前節の自然さの意味付け沿ってより直観的に理解するなら，次のようになるだろう．**1** の解釈に注目する．**1** には X の情報は含まれていないが，もしこれを BDD とみたうえで組合せ集合として解釈すれば，結果は $\mathcal{P}(X)$ となり，X が何であるかに依存する．また，**1** を ZDD とみると解釈は $\{\varepsilon\}$ になるが，これはブール関数として表現しようとすれば $\bigwedge_{x\in X}(\neg x)$ のようになり，やはり X が何であるかに依存する．このような，ダイアグラムには現れないにもかかわらず考慮しなければならない要素の情報が，「余分な情報」にあたると考えられそうである．

4 要素間の順序を考慮する場合

これまでは全体集合 X には何の構造も仮定しなかったが，実際に BDD や ZDD を扱う際は，あらかじめ要素間に順序を導入しておき，その順序に従って要素が出現するようなグラフを考える [4]（そのような BDD を特に ordered BDD (OBDD) と呼ぶこともある）．そのような制約を考慮しても，ダイアグラムの解釈は 3 節と同様に自然であることが示せる．ただし，その主張内容を正しく述べるには，関手と自然変換の定義を少し変更する必要がある．その変更に必要な用語の定義を導入する前に，本節の概要を述べる．

$(X, \leq)$ を全順序集合とする．X 上のダイアグラム F が $\leq$ と**整合する**とは，F において y が x の子孫として現れるならば $x < y$ であることとする．例えば図 1 のダイアグラムは $a \leq b$ となるとき，かつそのときに限り $\leq$ と整合する．

$\mathcal{D}'(X, \leq)$ を $\leq$ と整合する X 上のダイアグラム全体の集合とする．$\mathcal{D}'$ を以前と同じようにして関手にしたい．ところが，任意の関数を作用させることを考えると，不都合なことがある．というのは，もし $F \in \mathcal{D}'(X, \leq)$ に一般の関数 $f\colon X \to Y$ を以前と同じように作用させると，得られるダイアグラムは Y の順序と整合するとは限らないからである．例えば，$X = \{a, b\}$ とし，$f\colon X \to X$ を a と b を入れ替える関数と

すれば，図 1 のダイアグラムに f を作用させた結果は $\leq$ と整合しない．このような理由により，作用させる関数は任意の関数ではなく，狭義単調なものに限らなければならない．

この変更に合わせて，自然変換の定義も，「すべての狭義単調な関数について作用が保存されること」と変更する．以上の修正を施すと，定理 5 と同様の主張が $\mathcal{D}'$ について証明できる．

以上が順序が入っている場合の議論の概要である．これをもう少し丁寧に書き下すために，前節では明示的に現れなかった圏の概念を導入する．具体的な定義はきちんと書き下してみると少々長いのだが，基本的には（本稿では）関手の定義域を適切に制限するために導入するので，ある条件を満たす集合（対象）のクラスとその条件に合わせた関数（射）のクラスを指定するものだと考えておけばよい．

定義 7. **圏** $\mathscr{C}$ は，次のようなデータからなる．

- **対象** (object) と呼ばれる数学的対象の集まりが定まっている．X が $\mathscr{C}$ の対象であるとき，$X \in \mathscr{C}$ と書く（$\mathscr{C}$ は集合とは限らないが，そのことは本稿では問題にしない）．
- 各 $A, B \in \mathscr{C}$ に対して集合 $\mathscr{C}(A, B)$ が与えられている．$\mathscr{C}(A, B)$ の要素を（A から B への）**射** (arrow, morphism) と呼ぶ．考えている圏 $\mathscr{C}$ が明らかなときは，$f \in \mathscr{C}(A, B)$ であることを $f\colon A \to B$ とも書く．
- 各 $A, B, C \in \mathscr{C}$ と $f \in \mathscr{C}(A, B)$, $g \in \mathscr{C}(B, C)$ に対して，射 $g \circ f \in \mathscr{C}(A, C)$ が定まっている．これを f と g の**合成** (composition) と呼ぶ．
- 各対象 $A \in \mathscr{C}$ について，**恒等射** (identity) と呼ばれる射 $1_A \in \mathscr{C}(A, A)$ があり，すべての $B \in \mathscr{C}$ とすべての $f \in \mathscr{C}(A, B)$, $g \in \mathscr{C}(B, A)$ に対して $f \circ 1_A = f$, $1_A \circ g = g$ が成り立つ．
- 各 $A, B, C, D \in \mathscr{C}$ と $f \in \mathscr{C}(A, B)$, $g \in \mathscr{C}(B, C)$, $h \in \mathscr{C}(C, D)$ に対して $h \circ (g \circ f) = (h \circ g) \circ f$ が成り立つ（**結合律**）．

最も基本的な圏の一つは，集合と関数の圏 **Sets** であり，次のように定義される．

- **Sets** の対象は集合，つまり $A \in \mathbf{Sets}$ とは A が集合であることを意味する．
- **Sets** の射は関数である．つまり，$f \in \mathbf{Sets}(A, B)$ は f が A から B への関数であることを意味する．
- 恒等射は恒等写像，合成は通常の関数合成である．

このとき，よく知られている通り，恒等射と合成が満たすべき条件が成立するので **Sets** は確かに圏である．以上のことを「集合と関数の全体は圏をなす」などと表現する．

同様にして，全順序集合と狭義単調関数の全体は圏をなすことが容易に確かめられる．本稿ではこの圏を **TO** と書く．**TO** の定義を正確に書き下すと次のようになる．

- **TO** の対象は全順序集合である．
- **TO** の射は狭義単調関数（$x < y$ ならば $f(x) < f(y)$ であるような f）である．
- 恒等射と合成は **Sets** と同様に定義する．

恒等射は狭義単調だから確かに **TO** の射になる．狭義単調関数は関数合成に関して閉じているので，関数合成は **TO** における射の合成として well-defined である．恒等射の条件と結合律に関しては **Sets** と同様に成り立つ．よって **TO** は確かに圏になる．この圏 **TO** は，**Sets** とともに本節の議論において後で必要になる．

以上の例はいずれも対象が集合になっているが，一般には対象は集合とは限らない．実際，次節で考える vtree の圏における対象は集合ではなく木である（厳密に言えば，全順序集合も $(X, \leq)$ の形をした組だから集

合ではないともいえるが).

次に，一般の圏の間の関手を定義する．関手の概念は，定義 3 では集合と関数に対する操作として導入したが，ここでは圏の対象と射に関するものに一般化する．

定義 8. $\mathscr{C}, \mathscr{D}$ を圏とする．すべての対象 $X \in \mathscr{C}$ に対して対象 $\mathcal{F}(X) \in \mathscr{D}$ が与えられていて，かつすべての射 $f \in \mathscr{C}(X,Y)$ に対して射 $\mathcal{F}(f) \in \mathscr{D}(\mathcal{F}(X), \mathcal{F}(Y))$ が与えられているとする．$\mathcal{F}$ が $\mathscr{C}$ から $\mathscr{D}$ への**関手** $\mathcal{F}$ であるとは，以下の条件が成り立つことである．

- f が恒等射ならば $\mathcal{F}(f)$ も恒等射である．
- すべての $\mathscr{C}$ の射 f, g について，$f \circ g$ が定義されていれば $\mathcal{F}(f \circ g) = \mathcal{F}(f) \circ \mathcal{F}(g)$ である．

上の定義において $\mathscr{C} = \mathscr{D} = \mathbf{Sets}$ の場合を考えると，定義 3 に一致することがわかるだろう．定義 3 では f の作用 $F(f)$ は関数だったが，一般の圏においては作用は行き先の圏における射で表される．$\mathcal{F}$ が $\mathscr{C}$ から $\mathscr{D}$ への関手であることを，関数や射と同様に $\mathcal{F}\colon \mathscr{C} \to \mathscr{D}$ で表す．

関手のうち最も簡単なものの一つは（恒等関手を除けば）忘却関手だろう．忘却関手にもいろいろなバリエーションがあるが，基本的には対象の持っている情報の一部を「忘れる」関手のことを指す．本稿で必要になるのは既に導入した圏 **TO** から **Sets** への忘却関手 $|\cdot|\colon \mathbf{TO} \to \mathbf{Sets}$ であり，これは直観的には全順序集合のもつ順序構造のことを忘れてただの集合とみなす関手である．

- 全順序集合 $(X, \leq)$ に対して，$|(X, \leq)| = X$ である．
- 狭義単調関数 $f\colon (X, \leq_X) \to (Y, \leq_Y)$ に対して，$|f| = f$ である．

TO の射であるところの狭義単調関数 f は，もともと関数なのだから，それはそのまま **Sets** の射になる．**Sets** の射であるためには，単調かどうかに関わらずとにかく関数でありさえすればよいからである（そもそも **Sets** の対象は順序の入っていないただの集合だから，**Sets** の射が狭義単調であるかどうかを考えること自体が一般には意味をなさないのだが)．上で定義した $|\cdot|$ が関手の定義を満たしていることは簡単に確かめることができる．

本節で重要なもう一つの関手が，上で導入した $\mathcal{D}'$ である．これも **TO** から **Sets** への関手になる．

- 全順序集合 $(X, \leq)$ に対して，$\mathcal{D}'(X, \leq)$ は上で定義した通り，$\leq$ と整合する X 上のダイアグラム全体の集合である．
- $\mathcal{D}'(f)$ は前節で定義した $\mathcal{D}(f)$ と同じである．

$\mathcal{D}'(f)$ の定義において，f は狭義単調関数のみをとることに注意すると，$\mathcal{D}'(f)$ は整合性を保つことが証明できるので，$f\colon (X, \leq_X) \to (Y, \leq_Y)$ とすると確かに $\mathcal{D}'(f)\colon \mathcal{D}'(X, \leq_X) \to \mathcal{D}'(Y, \leq_Y)$ となっており，$\mathcal{D}'$ は関手として well-defined であることがわかる．

自然変換の定義も次のように一般化される．

定義 9. $\mathscr{C}, \mathscr{D}$ を圏，$\mathcal{F}, \mathcal{G}$ を $\mathscr{C}$ から $\mathscr{D}$ への関手とする．すべての対象 $X \in \mathscr{C}$ に対して $\mathscr{D}$ の射 $\alpha_X\colon \mathcal{F}(X) \to \mathcal{G}(X)$ が与えられていて，すべての $\mathscr{C}$ の射 $f\colon X \to Y$ に対して $\mathcal{G}(f) \circ \alpha_X = \alpha_Y \circ \mathcal{F}(f)$ が成り立つとき，α は $\mathcal{F}$ から $\mathcal{G}$ への**自然変換**であるという．

こちらも $\mathscr{C} = \mathscr{D} = \mathbf{Sets}$ の場合を考えると，定義 4 に一致することをみるのは易しいだろう．一般の圏においては，対象は集合とは限らないから要素をとることはできない（しかも，対象が集合であっても射が関

数とは限らない）ので，作用の保存を以前のように $f \cdot \alpha(x) = \alpha(f \cdot x)$ の形に書くことはできない．しかし $\mathcal{G}(f) \circ \alpha_X = \alpha_Y \circ \mathcal{F}(f)$ と書いておけばどんな圏においても通用する．そのためはじめからこういう書き方をしておいたのである．

以上の準備のもとで，β, ζ はそれぞれ $\mathcal{D}'$ から BF と CS への自然変換になる，と言いたいところだが（そしてだいたい正しいのだが），このような主張をするのは厳密には問題がある．というのは，そもそも $\mathcal{D}'$ と BF や CS では関手の定義域が異なっており，それらの間の自然変換は定義されないためである．ここでは $\mathcal{D}'$ の定義域が **TO** なので，BF と CS もそれに合わせなければならない．さらにもう一つ細かいことを言うと，β や ζ は集合で添字付けられた関数の族だったが，今回は全順序集合で添字付けられた族にしなければならない．よって，これらのずれを調整して，正式には次のように述べることになる．

定理 10. $\beta'_{(X,\leq)} = \beta_X$, $\zeta'_{(X,\leq)} = \zeta_X$ で β', ζ' を定義する．

1. β' は $\mathcal{D}'$ から $BF \circ |\cdot|$ への自然変換である．
2. ζ' は $\mathcal{D}'$ から $CS \circ |\cdot|$ への自然変換である．

ただし $\circ$ は関手の合成を表し，$(BF \circ |\cdot|)(X, \leq) = BF(X)$, $(BF \circ |\cdot|)(f) = BF(f)$ のように定義される．

細かいところを気にするとこのような主張になるのだが，結局のところこれが言っているのは，すべての狭義単調関数 $f\colon X \to Y$ に対して

$$\beta_Y \circ \mathcal{D}'(f) = BF(f) \circ \beta_X, \quad \zeta_Y \circ \mathcal{D}'(f) = CS(f) \circ \zeta_X$$

という等式が成り立つということである．しかも $\mathcal{D}'(f) = \mathcal{D}(f)$ なので（正確には両辺では定義域が異なり，$\mathcal{D}'(f)$ は $\mathcal{D}(f)$ の $\mathcal{D}'(X, \leq)$ への制限だが），上の定理は以前に示した定理 5 からすぐに出てくる．いろいろとややこしいことを言っているのは，形式上の辻褄を合わせるためである．

5 Sentential Decision Diagrams

ここまでは BDD と ZDD の対比を見てきたが，両者の拡張である *sentential decision diagram (SDD)* [5] と *zero-suppressed SDD (ZSDD)* [11] に対しても同様のことがいえる．本節ではこれについて議論する．

5.1 制約のない SDD/ZSDD

SDD の定義は以下の通りである．本来は X だけではなく，vtree [12] と呼ばれる木を考えるが，これについては後ほど考察する．また，定義中の $p_1, \ldots, p_n$ に対してそれらが分割 (partition) になるという条件が本来は付くが，ひとまずこの条件は外して考えることにする．

定義 11. 集合 X が与えられたとき，その上の SDD α とそのブール関数としての解釈 $\sigma(\alpha) \in BF(X)$ は以下のように定義される．

- $\top$ と $\bot$ は SDD であり，その解釈はそれぞれ *true* と *false* である．
- $x \in X$ に対して，x と $\neg x$ は SDD であり，その解釈はそれぞれブール関数としての x と $\neg x$ である．
- p_i, s_i が $1 \leq i \leq n$ に対して SDD であるとき，$\{(p_1, s_1), \ldots, (p_n, s_n)\}$ は SDD であり，その解釈は $\bigvee_{i=1}^{n} (\sigma(p_i) \wedge \sigma(s_i))$ である．

定義はこれで全てであるが，この定義の背景について簡単に触れておく．SDD は，ブール関数の（適当な条件を満たす）分解 (decomposition) [13] のグラフ表現を意図して設計されている [5]．X, Y を変数の集合で $X \cap Y = \emptyset$ となるものとし，$X \cup Y$ 上のブール関数 f を論理式として表示することを考える．一般に，$f(X,Y) = \bigvee_{i=1}^{n}(p_i(X) \wedge s_i(Y))$ の形に表示することはどんな f に対しても可能だが，このような表示を f の (X,Y)-分解という．このとき p_i を**プライム** (prime) といい，s_i を**サブ** (sub) という．SDD は，ブール関数にこの分解を再帰的に施して得られる表示に対応する．分解を考える際には，変数の集合の分け方を指定する必要があり（上の X と Y），それを指定するのに vtree と呼ばれる木を用いるが，これについては後ほど導入する．

上では p_i について何の条件も課さなかったが，特に $i \neq j$ ならば $p_i \wedge p_j = \mathit{false}$，かつ $p_1 \vee \cdots \vee p_n = \mathit{true}$ であるとき，$p_1, \ldots, p_n$ は**分割** (partition) であるという．SDD の本来の定義 [5] では，プライムはこの条件を満たすことが要求されている．このとき，$f(X,Y) = \bigvee_{i=1}^{n}(p_i(X) \vee s_i(Y))$ と表示した場合，具体的な X, Y の値 x, y に対して $p_i(x) = 1$ となる i は一意的に決まるので，この i に対して $f(x,y) = s_i(y)$ となる．特に X が単一の変数で，$n = 2$, $p_1 = X$, $p_2 = \neg X$ の場合が Shannon 分解として知られるもので，上の分解はその一般化になっている．*Sentential* decision diagram という名前の由来は，このようにプライムにリテラルとは限らない一般の論理式 (sentence) を許していることに由来する [5].

BDD に対して ZDD を考えたのと同様に，ゼロサプレス型の SDD である ZSDD を考えることができ，それは以下のように定義される．

定義 12. 集合 X が与えられたときに，その上の ZSDD α とその組合せ集合としての解釈 $\xi(\alpha)$ は以下のように定義される．

- $\bot$ と ε は ZSDD であり，その解釈はそれぞれ $\emptyset$ と $\{\varepsilon\}$ である．
- $x \in X$ に対して，x と $\pm x$ は ZSDD であり，その解釈はそれぞれ $\{\{x\}\}$ および $\{\varepsilon, \{x\}\}$ である．
- p_i, s_i が $1 \leq i \leq n$ に対して ZSDD であるとき，$\{(p_1, s_1), \ldots, (p_n, s_n)\}$ は ZSDD であり，その解釈は $\bigcup_{i=1}^{n}(\xi(p_i) \sqcup \xi(s_i))$ である．ただし演算 $\sqcup$ は次のように定義される:

$$P \sqcup Q = \{A \cup B \mid A \in P, B \in Q\}.$$

上で定義された解釈について，前節の定理 5 と同様の定理を証明したい．そのために，SDD, ZSDD の集合をとる関手 *SDD*, *ZSDD* を作る．

定義 13. $\mathit{SDD}(X)$, $\mathit{ZSDD}(X)$ はそれぞれ X 上の SDD, ZSDD の集合とする．関数 $f\colon X \to Y$ に対して，その SDD/ZSDD への作用をラベルの付け替えで定義する．すなわち，

$$\mathit{SDD}(f)(\top) = \top, \quad \mathit{SDD}(f)(\bot) = \bot, \quad \mathit{SDD}(f)(x) = f(x), \quad \mathit{SDD}(f)(\neg x) = \neg f(x),$$
$$\mathit{SDD}(\{(p_i, s_i)\}_{i=1}^{n}) = \{(\mathit{SDD}(f)(p_i), \mathit{SDD}(f)(s_i))\}_{i=1}^{n}$$

である．$\mathit{ZSDD}(f)$ も同様とする．このとき *SDD*, *ZSDD* は（**Sets** から **Sets** への）関手になる．

これらの関手に対して，以下が成り立つ．証明は，$\mathit{BF}(f)$ がブール演算を保つことおよび $\mathit{CS}(f)$ が $\bigcup$ と $\sqcup$ を保つことに注意すればできる．

定理 14. 1. σ は *SDD* から *BF* への自然変換である．
2. ξ は *ZSDD* から *CS* への自然変換である．

SDD と ZSDD についても 4 節と同様の議論ができる．4 節の議論に現れる全順序を vtree に置き換え，適切に定義を変更すればよい．Vtree とは，集合の分割のしかたを表す二分木で，次のように定義される．

定義 15. 集合 X の *vtree* とは，根付き完全二分木であって，その葉の集合と X との間に一対一の対応が与えられているもののことである．

以下，$x \in X$ と対応付けられる葉を単に x，左右の子がそれぞれ v, w である vtree を (v, w) と書く．また，二分木 v が X の vtree であるとき，X を $|v|$ と書く．

5.2 Vtree に整合する SDD/ZSDD

全順序に対してそれと整合するダイアグラムを考えたように，vtree に対してもそれと整合する SDD/ZSDD を定義したい．これは，本稿では以下のように定義する．

定義 16. SDD/ZSDD の vtree との整合性を次のように再帰的に定義する（SDD と ZSDD で定義はほぼ同じなので，まとめて定義した）．

1. $\top$, $\bot$, ε は任意の vtree と整合する．
2. x, $\neg x$, $\pm x$ は $x \in |v|$ となる任意の v と整合する．
3. α が v または w と整合するなら，α は (v, w) と整合する．
4. $\{(p_1, s_1), \ldots, (p_n, s_n)\}$ は，各 p_i が v と，各 s_i が w と整合するなら，(v, w) と整合する．

この整合性の定義は SDD に関する既存の文献に現れる “α respects v” のそれと似ているが，同じではない．Darwiche による”α respects v” の定義 [5, Def. 5] においては，規則 2 の v は葉でなければならず，また規則 3 は含まれない．また Bova による定義 [3] には，上の規則 1, 2, 4 のみが含まれ，やはり規則 3 は含まれない．

上の定義から容易に導けることとして，次の主張が成り立つ．

命題 17. SDD $\alpha = \{(p_i, s_i)\}_{i=1}^{n}$ が (v_1, v_2) と整合するとき，$\sigma(\alpha)$ は $(|v_1|, |v_2|)$-分解を与える．

また，この命題の状況では，p_i は v_1 と，s_i は v_2 と整合するから，α に含まれる各 SDD についてもその解釈は分解を与えていることが帰納的にわかる．

次に，整合する SDD/ZSDD の集合をとる操作が適当な圏から **Sets** への関手になることをいいたい．そのために，まずそれらの関手の定義域になる，vtree の圏を定義する．この圏の射となる概念を次に定義する．

定義 18. v, w を vtree とする．関数 $f\colon |v| \to |w|$ が v から w への**埋め込み**であることを次のように再帰的に定義する．

- v が葉なら f は埋め込みである．
- $w = (w_1, w_2)$ であって，f は v から w_1 または w_2 への埋め込みならば，f は v から w への埋め込みである．
- $v = (v_1, v_2)$, $w = (w_1, w_2)$ であって，f は $|v_i|$ に制限すると v_i から w_i への埋め込みになる $(i = 1, 2)$ なら，f は v から w への埋め込みである．

Vtree とその間の埋め込み全体のなす圏を **VTree** と書く．すなわち，**VTree** の対象は vtree であり，

$\mathbf{VTree}(v, w)$ は v から w への埋め込み全体の集合である．これが圏になっていることを確かめるのは，埋め込みが合成で閉じていることを示すのにいくつか場合分けが必要だが，難しくはない．

このように定義すると，vtree の埋め込み f に対し，先に定義した f の SDD/ZSDD への作用は整合性を保つ．よって，vtree v に対してそれと整合する SDD の全体 $SDD'(v)$ をとる関手 SDD'，および同様にして関手 $ZSDD'$ が定義できる．これらの関手の定義域は $\mathbf{VTree}$ である．念のため，その定義をまとめておく．

- $SDD'(v)$ は v と整合する SDD 全体，$ZSDD'(v)$ は v と整合する ZSDD 全体の集合である．
- 埋め込み $f\colon v \to w$ に対して，$SDD'(f)$ と $ZSDD'(f)$ はいずれもすべての葉 $x, \neg x, \pm x$ をそれぞれ $f(x), \neg f(x), \pm f(x)$ に書き換える操作である．

定理 19. $\sigma'_v = \sigma_{|v|}, \xi'_v = \xi_{|v|}$ とする．また関手 $U\colon \mathbf{VTree} \to \mathbf{Sets}$ を $U(v) = |v|, U(f) = f$ で定める．

1. σ' は SDD' から $BF \circ U$ への自然変換である．
2. ξ' は $ZSDD'$ から $CS \circ U$ への自然変換である．

証明は単純に計算すれば帰納法でできるが，より抽象的な証明を，圏論に慣れている読者のために記しておく．SDD' は $SDD \circ U$ の部分関手であることは容易に確認でき，$i\colon SDD' \to SDD \circ U$ を埋め込みとすれば $\sigma' = \sigma U \circ i$ と書ける．そうすると，定理 14 より $\sigma\colon SDD \to BF$ は自然であるから，σ' が自然であることはすぐにわかる．もちろん ξ' についても同様である．この証明は定理 10 にも適用できる．

5.3 プライムが分割になっている SDD/ZSDD

SDD の本来の定義 [5] では，プライムは分割になっていることが要求されていた．ここではこの制約が関数の作用で保存されるかどうかを考察する．

定義 20. α を SDD とする．

1. α が**部分分割** *SDD* とは，α のすべての節点 $\{(p_i, s_i)\}_{i=1}^n$ について，そのプライムが SDD として互いに素である，すなわち $i \neq j$ ならば $\sigma(p_i) \wedge \sigma(p_j) = \mathit{false}$ であることをいう．
2. α が**分割** *SDD* とは，α が部分分割 SDD であって，かつそのすべての節点 $\{(p_i, s_i)\}_{i=1}^n$ に対して $\sigma(p_1) \vee \cdots \vee \sigma(p_n) = \mathit{true}$ が成り立つことをいう．

定義 21. ZSDD α が**部分分割** *ZSDD* とは，α のすべての節点について，そのプライムが ZSDD として互いに素，すなわち $i \neq j$ ならば $\xi(p_i) \cap \xi(p_j) = \emptyset$ であることをいう．

ここで注意してほしいのは，上に定義した概念はいずれも vtree に無関係であるということである．一方で，分割 ZSDD という概念を定義しようとすると，どの vtree で考えるかを指定しなければならなくなる．というのは，ZSDD は組合せ集合として解釈されるので，同じ ZSDD でもどういう変数の集合を考えているかによって分割であるか否かが変わるのである．例として $\alpha = \{(a, \varepsilon), (\varepsilon, b)\}$ という ZSDD を考えてみよう．この ZSDD α のプライムはそれぞれ $\{a\}, \{\varepsilon\}$ という組合せ集合で解釈される．vtree (a, b) を考えると，この vtree の左の子に現れるのは a のみだから，α のプライムは分割を与える．一方，$((a, c), b)$ を考えると，左の子に c が現れるので，この vtree に対して α のプライムは分割になっていない．

この考察からすぐにわかることは，仮に分割 SDD と同様にして分割 ZSDD を（vtree v に相対的に）定義したとしても，埋め込みの作用は分割 ZSDD を保存しないということである．実際，上で考えた α と (a, b)

から $((a,c),b)$ への埋め込みが反例を与えている．

一方，その他の三つは保たれる（部分分割 ZSDD は単射性が必要）ことが証明できる．特に vtree の埋め込みは単射なので，これらの概念はすべて vtree の埋め込みで保たれることがわかる．

補題 22. $f\colon X \to Y$ を関数とする．

1. $SDD(f)$ は分割 SDD と部分分割 SDD を保存する．
2. f が単射ならば，$ZSDD(f)$ は部分分割 ZSDD を保存する．

Proof. 帰納法で証明する．

1. $\alpha = \{(p_i, s_i)\}_{i=1}^n \in SDD(X)$ とする．まず，$\{p_i\}_i$ が互いに素ならば $\{SDD(f)(p_i)\}_i$ も互いに素であることを示す．$\{p_i\}_i$ が互いに素で $i \neq j$ と仮定して，$\sigma(SDD(f)(p_i)) \wedge \sigma(SDD(f)(p_j)) = \mathit{false}$ をいえばよい．これは σ が自然変換であることと $\sigma(p_i) \wedge \sigma(p_j) = \mathit{false}$ であることを用いれば

$$\begin{aligned}\sigma(SDD(f)(p_i)) \wedge \sigma(SDD(f)(p_j)) &= BF(f)(\sigma(p_i)) \wedge BF(f)(\sigma(p_j)) \\ &= BF(f)(\sigma(p_i) \wedge \sigma(p_j)) \\ &= \mathit{false}\end{aligned}$$

   と確かめられる．ここで $BF(f)$ は論理式に対する操作としては命題変数への代入に相当しており，それは論理的同値性を保つことを利用している．同様にして $\bigvee_i \sigma(p_i) = \mathit{true}$ ならば $\bigvee_i \sigma(SDD(f)(p_i)) = \mathit{true}$ であることも示せる．よって $SDD(f)$ は分割 SDD および部分分割 SDD を保存する．
2. $\alpha = \{(p_i, s_i)\}_{i=1}^n \in ZSDD(X)$ とする．$\{p_i\}_i$ が互いに素ならば $\{ZSDD(f)(p_i)\}_i$ も互いに素であることを示せばよい．$\{p_i\}_i$ が ZSDD として互いに素で $i \neq j$ とすると，$\xi(p_i) \cap \xi(p_j) = \emptyset$ である．このとき $\xi(ZSDD(f)(p_i)) \cap \xi(ZSDD(f)(p_j)) = \emptyset$ をいえばよい．ξ が自然変換であることを用いると，これは $CS(f)(\xi(p_i)) \cap CS(f)(\xi(p_j)) = \emptyset$ と同値であることがわかる．左辺が空であることをいうためには，

$$B \in CS(f)(\xi(p_i)) \cap CS(f)(\xi(p_j)) \iff \exists A \in \xi(p_i), \exists A' \in \xi(p_j), B = f(A) = f(A')$$

   なので，$A \in \xi(p_i), A' \in \xi(p_j)$ に対して $f(A) \neq f(A')$ を示せば十分である．ところが，f が単射であれば f による像をとる関数 $\mathcal{P}(X) \to \mathcal{P}(Y)$ も単射である．よって $f(A) \neq f(A')$ が成り立つためには $A \neq A'$ であれば十分だが，仮定より $\xi(p_i) \cap \xi(p_j) = \emptyset$ であるから $A \neq A'$ は明らかに成り立つ． □

この補題より，分割 SDD や部分分割 SDD/ZSDD の集合をとる関手を定義することができる．また vtree と整合的な分割 SDD などに対しても同様である．それらの関手に対して定理 19 と同様のことが成り立ち，証明も全く同様にできる．

この考察から，分割と互いに素という概念は，SDD に対してはいずれも相性がよいが，分割と ZSDD の相性はよくないのではないかということが予想される．ZSDD を導入した論文 [11] では，通常の ZSDD の定義としては上で定義した分割 ZSDD が採用されているが，論文の後半で implicit partition という考え方を導入している．彼らはそこで ZSDDs with implicit partition という別の種類の ZSDD を導入しており，これはわれわれが上で定義した部分分割 ZSDD に含まれる．われわれが上で行った考察の結果は，ZSDD の取り扱いにおいて implicit partition という考え方が自然なものであることを示唆しているように思われる．

参考文献

[1] Sheldon B. Akers. Binary decision diagrams. *IEEE Transactions on Computers*, Vol. C-27, No. 6, pp. 509–516, June 1978.

[2] Steve Awodey. *Category Theory*. Oxford Logic Guides. Oxford University Press, 2010.

[3] Simone Bova. SDDs are exponentially more succinct than OBDDs. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, pp. 929–935. AAAI Press, 2016.

[4] Randal E. Bryant. Graph-based algorithms for boolean function manipulation. *IEEE Trans. Comput.*, Vol. 35, No. 8, pp. 677–691, August 1986.

[5] Adnan Darwiche. SDD: A new canonical representation of propositional knowledge bases. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence - Volume Volume Two*, IJCAI'11, pp. 819–826. AAAI Press, 2011.

[6] Samuel Eilenberg and Saunders Mac Lane. Natural isomorphisms in group theory. In *Proceedings of the National Academy of Sciences of the United States of America*, Vol. 28, pp. 537–543, 1942.

[7] Samuel Eilenberg and Saunders Mac Lane. General theory of natural equivalences. *Transactions of the American Mathematical Society*, Vol. 58, No. 2, pp. 231–294, September 1945.

[8] Saunders Mac Lane and Ieke Moerdijk. *Sheaves in Geometry and Logic*. Springer, 1992.

[9] C. Y. Lee. Representation of switching circuits by binary-decision programs. *The Bell System Technical journal*, Vol. 38, No. 4, pp. 985–999, July 1959.

[10] Shin-ichi Minato. Zero-suppressed BDDs for set manipulation in combinatorial problems. In *Proceedings of the 30th International Design Automation Conference*, DAC '93, pp. 272–277, New York, NY, USA, 1993. ACM.

[11] Masaaki Nishino, Norihito Yasuda, Shin-ichi Minato, and Masaaki Nagata. Zero-suppressed sentential decision diagrams. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, pp. 1058–1066. AAAI Press, 2016.

[12] Knot Pipatsrisawat and Adnan Darwiche. New compilation languages based on structured decomposability. In *Proceedings of the 23rd National Conference on Artificial Intelligence - Volume 1*, AAAI'08, pp. 517–522. AAAI Press, 2008.

[13] Knot Pipatsrisawat and Adnan Darwiche. A lower bound on the size of decomposable negation normal form. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence*, AAAI'10, pp. 345–350. AAAI Press, 2010.

[14] 戸田貴久, 斎藤寿樹, 岩下洋哲, 川原純, 湊真一. ZDD と列挙問題—最新の技法とプログラミングツール. コンピュータ ソフトウェア, Vol. 34, No. 3, pp. 3_97–3_120, 2017.

[15] 小島健介. ブール関数と組合せ集合の圏論的性質に基づく BDD と ZDD の比較. *SIG-FPAI*, Vol. B5, No. 03, pp. 57–62, March 2018.