Coq/SSreflect を用いた 条件付独立性の形式化について

山口 龍太郎 * 金 健 * 下山 周悟 *
萩原 学 * 山本 光晴 * 汪 金芳 *
* 千葉大学大学院理学研究科基盤理学専攻数学・情報数理学コース

概要. 本稿は，定理証明支援系 Coq とその拡張である SSReflect を用いた，統計学における条件付き独立性の形式化を目標とした．具体的に，Wang(2010) で提案された代数系 Cain の形式化を行った．証明支援系を用いた統計学関連の形式化は，世界的にも珍しい研究内容であり，条件付独立性の研究だけでなく，統計的因果推論の研究にも大きな影響を与えることが予想される．

Formalization of the Conditional Independence using Coq/SSReflect

Rutaro Yamaguchi* Ken Kin* Shugo Shimoyama*
Manabu Hagiwara* Mitsuharu Yamamoto* Jinfang Wang*
*Department of Mathematics and Informatics Graduate School of Science, Chiba University, Japan

Abstract. The past decade has witnessed dramatic progresses in formalization of mathematical theories using interactive theorem provers, such as Coq/SSReflect. These theories include the Four Color Theorem and the Feit-Thompson Theorem. In this paper, we use Coq/SSReflect, one of the most popular proof assistants, to formalize probabilistic conditional independence (PCI) relations in statistical sciences. Among the existing axiomatic systems of PCI, we have chosen Cain, an algebraic framework recently proposed by Wang (2010). The cain has advantages over other axiomatic systems in that all axioms and relations on PCI are expressed in equational forms. Our paper is among the first attempts on formalization of statistical theories. The formalization of PCI will be of importance to statistical inference in general, and to the studies of statistical causal inference in particular.

1. Cain Algebra

1.1 Cainoid

連続型の確率変数や条件付独立性について形式化していくため，まずこれらを代数的に表現したい．それに当たり，今回 Wang [1] が提唱した cain と呼ばれる代数構造を導入する．

二項関係 $\leq$ を持つ半順序集合 $\mathbb{L}$ を有界束とする．すなわち，任意の $x, y \in \mathbb{L}$ に対して上限（join）$x \vee y$ と下限（meet）$x \wedge y$ が存在し，かつ任意の x に対して $\emptyset \leq x$ となる $\emptyset$ と，$x \leq \top$となる$\top$ が存在する．$x \wedge y = \emptyset$ であるとき，x と y は互いに排反であるという．

$\mathbb{L}$ の直積 $\mathbb{L} \otimes \mathbb{L} = \{(x, y) \mid x, y \in \mathbb{L}\}$ について考える．これらは非対称であるので，その非対称性を強調するために，この直積の元を新しい記号 $\mathbb{T}^x_y$ を用いて表す．また，簡単のために，

$$\mathbb{T}^x = \mathbb{T}^x_\emptyset,\ \mathbb{T}_y = \mathbb{T}^\emptyset_y,\ \mathbb{T}^\emptyset_\emptyset = 1$$

と表記することにする（$\mathbb{T}^x$ を raising coin，$\mathbb{T}_x$ を lowering coin と呼ぶ.）．ここで使っている 1 は単位元という意味である．これらをまとめて atom coin と呼ぶ．atom coin をつなげたもの，つまり，$\mathbb{T} = \mathbb{T}^{x_1}_{y_1}\mathbb{T}^{x_2}_{y_2}\cdots\mathbb{T}^{x_n}_{y_n}$ を元として持つ集合

$$\mathfrak{C} = \left\{\mathbb{T}^{x_1}_{y_1}\mathbb{T}^{x_2}_{y_2}\cdots\mathbb{T}^{x_n}_{y_n} \mid x_i, y_i \in \mathbb{L},\ i = 1, \cdots, n,\ n \in \mathbb{N}\right\}$$

を，coin と呼ぶことにする．もちろん，atom coin も $\mathfrak{C}$ の元である．

定義 1.1 (cainoid) $\mathbb{L}$ を最小元 $\emptyset$ を持つ有界束とする．任意の $\mathbb{T}, \mathbb{T}', \mathbb{T}''$ と任意の $x, y \in \mathbb{L}$ に対して次が成り立つとき，$(\mathfrak{C}, \cdot)$ を cainoid と呼ぶ.

$$\mathbb{T} \cdot \mathbb{T}' = \mathbb{T}' \cdot \mathbb{T} \tag{1.1}$$

$$(\mathbb{T} \cdot \mathbb{T}') \cdot \mathbb{T}'' = \mathbb{T} \cdot (\mathbb{T}' \cdot \mathbb{T}'') \tag{1.2}$$

$$1 \cdot \mathbb{T} = \mathbb{T} \tag{1.3}$$

$$\mathbb{T}^x \cdot \mathbb{T}_x = 1 \tag{1.4}$$

$$\mathbb{T}^x_y = \mathbb{T}^{x\vee y} \cdot \mathbb{T}_y \quad (x > \emptyset) \tag{1.5}$$

これらの公理は，確率密度関数から導かれる最も基本的な性質とみなすことができる．例えば $\mathbb{T}^x$ は確率密度関数 $f(x)$，$\mathbb{T}_x$ は $\frac{1}{f(x)}$ としてみれば，(1.5) 式は

$$f(x|y) = \frac{f(x, y)}{f(y)}$$

という，条件付密度関数の定義式に対応している．この公理から，Bayes の定理を簡単に導くことができる．

定理 1.2 (Bayes の定理)

$$\mathbb{T}_y^x = \mathbb{T}_x^y \mathbb{T}^x \mathbb{T}_y \quad (x, y > \emptyset) \tag{1.6}$$

証明

$$\mathbb{T}_x^y \mathbb{T}^x \mathbb{T}_y = \left(\mathbb{T}^{x\vee y}\mathbb{T}_x\right)\mathbb{T}^x\mathbb{T}_y = \mathbb{T}^{x\vee y}\left(\mathbb{T}_x\mathbb{T}^x\right)\mathbb{T}_y = \mathbb{T}^{x\vee y}\cdot\mathbb{T}_y = \mathbb{T}_y^x$$

□

補題 1.3 $(\mathfrak{C}, \cdot)$ はアーベル群である.

証明　結合法則，単位元の存在，交換法則は Cainoid の定義 1.1 内で定義されている．したがって $\mathfrak{C}$ の任意の元に対する逆元の存在を確認すればよい．(1.3) 式より，$\mathbb{T} = \mathbb{T}^x$ または $\mathbb{T} = \mathbb{T}_x$ については互いに逆元の関係にある．$\mathbb{T} = \mathbb{T}_y^x$ で $x > \emptyset$ の時については，

$$\begin{aligned}\mathbb{T}_y^x\left(\mathbb{T}_{x\vee y}\mathbb{T}^y\right) &= \left(\mathbb{T}^{x\vee y}\mathbb{T}_y\right)\left(\mathbb{T}_{x\vee y}\mathbb{T}^y\right)\\ &= \left(\mathbb{T}^{x\vee y}\mathbb{T}_{x\vee y}\right)\left(\mathbb{T}_y\mathbb{T}^y\right)\\ &= 1\end{aligned}$$

より，$\mathbb{T}_y^x$ の逆元は $\mathbb{T}_{x\vee y}\mathbb{T}^y$ である．

今，$\mathbb{T} = \prod_{i=1}^n \mathbb{T}_{y_i}^{x_i}$ を $\mathfrak{C}$ 内の任意の元とし，$(\mathbb{T}_{y_i}^{x_i})'$ を $\mathbb{T}_{y_i}^{x_i}$ の逆元とすると，

$$\mathbb{T}\cdot\prod_{i=1}^n(\mathbb{T}_{y_i}^{x_i})' = \prod_{i=1}^n\left\{\mathbb{T}_{y_i}^{x_i}(\mathbb{T}_{y_i}^{x_i})'\right\} = 1$$

より，任意の元 $\mathbb{T}$ に対し，逆元が存在する．□

ここで新しく，$\mathbb{T}$ の逆元を $(\mathbb{T})^{-1}$ と表記することにすると，$\mathbb{T}_y^x = \mathbb{T}^{x\vee y}\mathbb{T}_y = \mathbb{T}^{x\vee y}(\mathbb{T}^y)^{-1}$ と表すことができ，以下の補題を得る．

補題 1.4 任意の Cainoid $\mathfrak{C}$ の元は raising coin のみで表記できる.

証明　$\mathfrak{C}$ の元は $\mathbb{T}_{y_i}^{x_i} = \mathbb{T}^{x\vee y}(\mathbb{T}^y)^{-1}$ のみで表記されるので明らか.　□

1.2　R-law と L-law

この節では，coin を用いて（条件付）独立を語る上で重要な 2 つの補題を紹介する．

補題 1.5 (raising-up law (R-law)) $\forall x > \emptyset, y, z \in \mathbb{L}$ について以下が成り立つ.

$$\mathbb{T}^{x\vee y} = \mathbb{T}_y^x\mathbb{T}^y \tag{1.7}$$

$$\mathbb{T}_z^{x\vee y} = \mathbb{T}_{y\vee z}^x\mathbb{T}^y \Leftrightarrow \mathbb{T}^{y\vee z} = \mathbb{T}^y\mathbb{T}^z \tag{1.8}$$

証明　(1.7) 式は (1.5) 式より明らか．(1.8) を証明する．まず，(1.7) 式より

$$\mathbb{T}^{x\vee y\vee z} = \mathbb{T}^{x\vee y}_{z}\mathbb{T}^{z} = \mathbb{T}^{x}_{y\vee z}\mathbb{T}^{y\vee z}$$

である．すると，Cainoid の公理を用いて

$$\begin{aligned}\mathbb{T}^{x\vee y}_{z} = \mathbb{T}^{x\vee y}_{z}(\mathbb{T}^{z}\mathbb{T}_{z}) = \left(\mathbb{T}^{x\vee y}_{z}\mathbb{T}^{z}\right)\mathbb{T}_{z} &= \left(\mathbb{T}^{x}_{y\vee z}\mathbb{T}^{y\vee z}\right)\mathbb{T}_{z} \\ &= \mathbb{T}^{x}_{y\vee z}(\mathbb{T}^{y}\mathbb{T}_{y})\left(\mathbb{T}^{y\vee z}\mathbb{T}_{z}\right) \\ &= \underbrace{\left(\mathbb{T}^{x}_{y\vee z}\mathbb{T}^{y}\right)}_{(a)}\underbrace{\left(\mathbb{T}_{y}\mathbb{T}^{y\vee z}\mathbb{T}_{z}\right)}_{(b)}\end{aligned}$$

と書き換えられる．(⇒) は (*a*) の部分が左辺と等しくなるので，

$$\begin{aligned}\mathbb{T}^{x\vee y}_{z} &= \mathbb{T}^{x\vee y}_{z}\left(\mathbb{T}_{y}\mathbb{T}^{y\vee z}\mathbb{T}_{z}\right) \\ \therefore\ \mathbb{T}_{y}\mathbb{T}^{y\vee z}\mathbb{T}_{z} &= 1 \\ \therefore\ \mathbb{T}^{y\vee z} &= \mathbb{T}^{y}\mathbb{T}^{z}\end{aligned}$$

(⇐) は (*b*) の部分の計算結果が単位元となるので

$$\mathbb{T}^{x\vee y}_{z} = \mathbb{T}^{x}_{y\vee z}\mathbb{T}^{y}$$

□

下付きの文字を上付きに持ってくるためには，raising coin をかければよいが，変数が 3 つになるとその等式は独立を表す等式と同値である．(1.8) で $z = \emptyset$ とすれば，(1.7) と同様の式が出てくる．この補題は，確率密度関数の表記にすれば，

$$f(x|y) = f(x,y)f(y)$$

$$f(x,y\,|\,z) = f(x|y,z)f(y) \Leftrightarrow f(y,z) = f(y)f(z)$$

のようになる．上付きの文字を下付きに持ってくる時にも同様な関係が成り立つ．

補題 1.6 (lowering-up law (L-law)) $\forall x > \emptyset, y, z \in \mathbb{L}$ について以下が成り立つ．

(1.9)
$$\mathbb{T}^{x}_{y} = \mathbb{T}^{x\vee y}\mathbb{T}_{y}$$

(1.10)
$$\mathbb{T}^{x}_{y\vee z} = \mathbb{T}^{x\vee y}_{z}\mathbb{T}_{y} \Leftrightarrow \mathbb{T}^{y\vee z} = \mathbb{T}^{y}\mathbb{T}^{z}$$

これらの証明は R-law の証明と同様の手順である．

1.3 canonical expression

cainoid の定義によると coin には様々な表記の仕方がある．例えば，$x_1 \vee x_2 = y_1 \vee y_2 \vee y_3$ とすると

$$\begin{aligned} \mathbb{T}^{x_1 \vee x_2} &= \mathbb{T}^{x_1} \mathbb{T}^{x_2}_{x_1} = \mathbb{T}^{x_2} \mathbb{T}^{x_1}_{x_2} \\ &= \mathbb{T}^{y_1} \mathbb{T}^{y_2}_{y_1} \mathbb{T}^{y_3}_{y_1 \vee y_2} = \mathbb{T}^{y_2} \mathbb{T}^{y_1}_{y_2} \mathbb{T}^{y_3}_{y_1 \vee y_2} \end{aligned}$$

というように表すことができる．ここではこういった coin の表現のうち扱いやすい形について考える．また，以下では $x, y \in \mathbb{L}$ に対して $x \neq y$ であるとき，$\mathbb{T}^x, \mathbb{T}^y$ は互いに異なるという．

定理 1.7 任意の $\mathbb{T}(\neq 1)$ に対して，

$$\mathbb{T} = (\mathbb{T}^{x_1})^{n_1} (\mathbb{T}^{x_2})^{n_2} \cdots (\mathbb{T}^{x_r})^{n_r} \tag{1.11}$$

となるような 0 でない整数 n_i と $\mathbb{T}^{x_i}$ が存在する．ただし，$i = 1 \cdots r$ であり，$\mathbb{T}^{x_i}$ は互いに異なる．

証明　任意の coin に対して，その性質より $\mathbb{T} = \mathbb{T}^{c_1}_{d_1} \mathbb{T}^{c_2}_{d_2} \cdots \mathbb{T}^{c_s}_{d_s}$ $(c_i \vee d_i \neq \emptyset)$ と表せる．

$$\mathbb{T}^{c_i}_{d_i} = \begin{cases} \mathbb{T}^{c_i \vee d_i} \mathbb{T}_{d_i} = \mathbb{T}^{c_i \vee d_i} (\mathbb{T}^{d_i})^{-1} & (c_i > \emptyset) \\ \mathbb{T}_{d_i} = (\mathbb{T}^{d_i})^{-1} & (c_i = \emptyset) \end{cases} \tag{1.12}$$

であるので，$e_i = c_i \vee d_i$, $m \geq 0$ とすれば

$$\mathbb{T} = \{\mathbb{T}^{e_1} (\mathbb{T}^{d_1})^{-1} \cdots \mathbb{T}^{e_m} (\mathbb{T}^{d_m})^{-1}\} \{(\mathbb{T}^{d_{m+1}})^{-1} \cdots (\mathbb{T}^{d_s})^{-1}\} \tag{1.13}$$

と表すことができる．今，

$$\{f_1, f_2, \cdots, f_t\} = \{e_1, d_1, \cdots, e_m, d_m, d_{m+1}, \cdots, d_s\} \tag{1.14}$$

とおけば $i \neq j$ に対して $f_i \neq f_j$ である．さらに，

$$\begin{aligned} u_i &= \#\{e_j = f_i \mid j = 1, \cdots, m\} \\ v_i &= \#\{d_j = f_i \mid j = 1, \cdots, s\} \end{aligned}$$

とすれば，式 (1.13) より

$$\begin{aligned} \mathbb{T} &= (\mathbb{T}^{f_1})^{u_1} (\mathbb{T}^{f_1})^{-v_1} \cdots (\mathbb{T}^{f_t})^{u_t} (\mathbb{T}^{f_t})^{-v_t} \\ &= (\mathbb{T}^{f_1})^{u_1 - v_1} \cdots (\mathbb{T}^{f_t})^{u_t - v_t} \\ &= (\mathbb{T}^{f_1})^{n_1} \cdots (\mathbb{T}^{f_t})^{n_t} \quad (n_i = u_i - v_i,\ n_i \neq 0) \end{aligned}$$

したがって，定理は成り立つ．　□

任意の coin に対する上記の定理のような表記に対し，各 coin の持つ束の元についてさらに以下のような関数を定義する．

定義 1.8 (context) 任意の $\mathbb{T}$ に対して，式 (2.1) のように分解されたとき，$x_1, x_2, \cdots, x_r$ を join でつなぎ合わせたもの，すなわち，$\vee_{i=1}^{r} x_i$ を $\mathbb{T}$ の context とよび，$\mathfrak{J}(\mathbb{T}) = \vee_{i=1}^{r}$ とかく．

定義 1.9 任意の $\mathbb{T}$ に対して，$\mathfrak{J}(\mathbb{T}) = x$ であるとき，$\mathbb{T}\{x\}$ を $\mathfrak{J}(\mathbb{T}\{x\}) = x$ として用いる．

補題 1.10 $\mathbb{T} \in \mathfrak{C}$ に対して

$$\mathfrak{J}(\mathbb{T}) = \mathfrak{J}(\mathbb{T}^{-1})$$

証明

$$\mathbb{T} = \left(\mathbb{T}^{x_1}\right)^{n_1} \cdots \left(\mathbb{T}^{x_r}\right)^{n_r}$$

で，定理 1.7 の性質をすべて満たすとすると，$\mathbb{T}^{-1}$ は

$$\mathbb{T}^{-1} = \left(\mathbb{T}^{x_1}\right)^{-n_1} \cdots \left(\mathbb{T}^{x_r}\right)^{-n_r}$$

と表記でき，これらの context は $\vee_{i=1}^{r} x_i$ である． □

補題 1.11 $\mathbb{T}, \mathbb{T}' \in \mathfrak{C}$ に対して

$$\mathfrak{J}(\mathbb{T}) \vee \mathfrak{J}(\mathbb{T}') \geq \mathfrak{J}(\mathbb{T}\mathbb{T}')$$

証明 $\mathbb{T}, \mathbb{T}'$ がそれぞれ定理 1.7 の性質をすべて満たすように，

$$\mathbb{T} = \left(\mathbb{T}^{x_1}\right)^{n_1} \cdots \left(\mathbb{T}^{x_r}\right)^{n_r}$$
$$\mathbb{T}' = \left(\mathbb{T}^{y_1}\right)^{m_1} \cdots \left(\mathbb{T}^{y_s}\right)^{m_s}$$

と表記できるとする．ここで，$\{z_1, \cdots, z_t\}$ を

$$\{z_1, \cdots, z_t\} = \{x_1, \cdots x_r, y_1, \cdots y_s\}$$

と定義する．z_i は x_j, y_k のいずれかと一致する．今，

$$k_i = \begin{cases} n_j & (z_i = x_j) \\ m_k & (z_i = y_k) \\ n_j + m_k & (z_i = x_j = y_k) \end{cases}$$

を用いると，$\mathbb{T}\mathbb{T}'$ は定理 1.7 の性質を満たすように

$$\mathbb{T}\mathbb{T}' = \left(\mathbb{T}^{z_1}\right)^{k_1} \cdots \left(\mathbb{T}^{z_w}\right)^{k_w} \quad (w \leq t)$$

と表記できる．従って，

$$\begin{aligned}\mathfrak{J}(\mathbb{T}\mathbb{T}') &= z_1 \vee \cdots z_w \\ &\leq (z_1 \vee \cdots z_w) \vee (z_{w+1} \vee \cdots z_t) \\ &= (x_1 \vee \cdots x_r) \vee (y_1 \vee \cdots y_s) \\ &= \mathfrak{J}(\mathbb{T}) \vee \mathfrak{J}(\mathbb{T}')\end{aligned}$$

□

補題 1.12 $\mathbb{T}, \mathbb{T}' \in \mathfrak{C}$ に対して

$$\mathfrak{J}(\mathbb{T}) \vee \mathfrak{J}(\mathbb{T}\mathbb{T}') = \mathfrak{J}(\mathbb{T}') \vee \mathfrak{J}(\mathbb{T}\mathbb{T}')$$

証明　補題 1.10，補題 1.11 より，

$$\mathfrak{J}(\mathbb{T}^{-1}) \vee \mathfrak{J}(\mathbb{T}\mathbb{T}') \geq \mathfrak{J}(\mathbb{T}^{-1}\mathbb{T}\mathbb{T}') = \mathfrak{J}(\mathbb{T}')$$

$$\therefore\ \mathfrak{J}(\mathbb{T}) \vee \mathfrak{J}(\mathbb{T}\mathbb{T}') \geq \mathfrak{J}(\mathbb{T}')$$

$$\therefore\ \mathfrak{J}(\mathbb{T}) \vee \mathfrak{J}(\mathbb{T}') \vee \mathfrak{J}(\mathbb{T}\mathbb{T}') = \mathfrak{J}(\mathbb{T}) \vee \mathfrak{J}(\mathbb{T}\mathbb{T}')$$

同様に，

$$\mathfrak{J}(\mathbb{T}'^{-1}) \vee \mathfrak{J}(\mathbb{T}\mathbb{T}') \geq \mathfrak{J}(\mathbb{T}'^{-1}\mathbb{T}\mathbb{T}') = \mathfrak{J}(\mathbb{T})$$

$$\therefore\ \mathfrak{J}(\mathbb{T}') \vee \mathfrak{J}(\mathbb{T}\mathbb{T}') \geq \mathfrak{J}(\mathbb{T})$$

$$\therefore\ \mathfrak{J}(\mathbb{T}) \vee \mathfrak{J}(\mathbb{T}') \vee \mathfrak{J}(\mathbb{T}\mathbb{T}') = \mathfrak{J}(\mathbb{T}') \vee \mathfrak{J}(\mathbb{T}\mathbb{T}')$$

以上より，

$$\mathfrak{J}(\mathbb{T}) \vee \mathfrak{J}(\mathbb{T}\mathbb{T}') = \mathfrak{J}(\mathbb{T}') \vee \mathfrak{J}(\mathbb{T}\mathbb{T}')$$

□

補題 1.13 任意の $x, y \in \mathbb{L}$ に対し，$\mathbb{T}$ が x 可積分 かつ $y \wedge x = \emptyset$ ならば $\mathbb{T}'\mathbb{T}$ もまた x 可積分 である．ただし，$\mathbb{T}'$ は $\mathfrak{J}(\mathbb{T}') \leq y$ を満たす．

証明　$\mathfrak{J}(\mathbb{T}') \leq y$ より

$$\mathfrak{J}(\mathbb{T}') \wedge x \leq y \wedge x = \emptyset$$

また，$\mathbb{T}$ が x 可積分であることから $\mathfrak{J}(\mathbb{T}) \wedge x = x$ である．よって，

$$\begin{aligned}\mathfrak{J}(\mathbb{T}'\mathbb{T}) \vee x &= \mathfrak{J}(\mathbb{T}'\mathbb{T}) \vee (\mathfrak{J}(\mathbb{T}) \wedge x) \\ &= (\mathfrak{J}(\mathbb{T}'\mathbb{T}) \vee \mathfrak{J}(\mathbb{T})) \wedge (\mathfrak{J}(\mathbb{T}'\mathbb{T}) \vee x) \\ &= (\mathfrak{J}(\mathbb{T}'\mathbb{T}) \vee \mathfrak{J}(\mathbb{T}')) \wedge (\mathfrak{J}(\mathbb{T}'\mathbb{T}) \vee x) \\ &= \mathfrak{J}(\mathbb{T}'\mathbb{T}) \vee (\mathfrak{J}(\mathbb{T}') \wedge x) \\ &= \mathfrak{J}(\mathbb{T}'\mathbb{T}) \vee \emptyset \\ &= \mathfrak{J}(\mathbb{T}'\mathbb{T}) \\ \therefore \mathfrak{J}(\mathbb{T}'\mathbb{T}) &\geq x\end{aligned}$$

ゆえに $\mathfrak{J}(\mathbb{T}'\mathbb{T})$ は x 可積分. □

1.4 cain

有界束 $\mathbb{L}$ の任意の元 x に対して，$x \vee \bar{x} = \top, x \wedge \bar{x} = \emptyset$ となるような補元 $\bar{x}$ が存在するとき $\mathbb{L}$ を可補束という．また，可補束 $\mathbb{L}$ が分配性をもつ，すなわち，任意の $\mathbb{L}$ の元 x, y, z に対して

$$x \vee (y \wedge z) = (x \vee y) \wedge (x \vee z)$$
$$x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z)$$

であるとき，補元は一意に定まる．以下では有界束 $\mathbb{L}$ を分配性をもつ可補束とする．

定義 1.14 (周辺化) $\mathbb{L}$ の任意の元 x に対して，x 可積分 である coin の集合を $\mathfrak{D}(x)$ とする．今，$\int_x$ を $\int_x : \mathfrak{D}(x) \to \mathfrak{C}$ であるような関数とし，$\mathfrak{D}(x)$ の任意の元 $\mathbb{T}\{y\}$ に対して

$$\int_x \mathbb{T}\{y\} = \mathbb{T}\{y \wedge \bar{x}\} \tag{1.15}$$

を定義する．また $\int_x$ は，以下の 3 つの性質を満たすとする．

(i) $\mathbb{T}^y$ が x 可積分 であるなら

$$\int_x \mathbb{T}^y = \mathbb{T}^{y \wedge \bar{x}} \tag{1.16}$$

(ii) $x_1 \wedge x_2 = \emptyset$ であるような x_1, x_2 に対して，$\mathbb{T}\{y_1\}, \mathbb{T}\{y_2\}$ を $\mathbb{T}\{y_1\}$ は x_1可積分，$\mathbb{T}\{y_2\}$ は x_2可積分 であるようにとり，$\mathbb{T}\{y_1\}\mathbb{T}\{y_2\}$ を $(x_1 \vee x_2)$ 可積分 とする．さらに，$x_1 \wedge y_2 = x_2 \wedge y_1 = \emptyset$ と仮定する．このとき，

$$\int_{x_1 \vee x_2} \mathbb{T}\{y_1\}\mathbb{T}\{y_2\} = \int_{x_1} \mathbb{T}\{y_1\} \int_{x_2} \mathbb{T}\{y_2\} \tag{1.17}$$

(iii) 任意の $\mathbb{T}$ に対して，

$$\int_\emptyset \mathbb{T} = \mathbb{T} \tag{1.18}$$

以上のような $\int_x$ の 3 つの性質 (1.16)~(1.18) を簡単のため次のような記法で表す．

$$\int \mathbb{T}^y dx = \mathbb{T}^{y \wedge \bar{x}} \tag{1.19}$$

$$\int \mathbb{T}\{y_1\}\mathbb{T}\{y_2\} d(x_1 \vee x_2) = \int \mathbb{T}\{y_1\} dx_1 \int \mathbb{T}\{y_2\} dx_2 \tag{1.20}$$

$$\int \mathbb{T} d\emptyset = \mathbb{T} \tag{1.21}$$

定義 1.15 (cain) (1.19)~(1.21) を満たす cainoid を *cain* とよぶ.

1.5 cain の性質

前節で定義した cain がどのような性質を持っているか，以下見ていく．

定理 1.16 $\mathbb{T}$ が x 可積分 であるとする．また，$x \wedge y = \emptyset$ かつ $\mathfrak{J}(\mathbb{T}') \leq y$ であるならば次が成り立つ．

(1.22)
$$\int \mathbb{T}' \mathbb{T} dx = \mathbb{T}' \int \mathbb{T} dx$$

証明 $\mathbb{T}'$ は $\emptyset$ 可積分なので

$$\mathbb{T}' \int \mathbb{T} dx = \int \mathbb{T}' d\emptyset \int \mathbb{T} dx$$

また，$\mathfrak{J}(\mathbb{T}') \leq y$ より $y \wedge \mathfrak{J}(\mathbb{T}') = \mathfrak{J}(\mathbb{T}')$ である．

$$\begin{aligned} \therefore x \wedge \mathfrak{J}(\mathbb{T}') &= x \wedge (y \wedge \mathfrak{J}(\mathbb{T}')) \\ &= (x \wedge y) \wedge \mathfrak{J}(\mathbb{T}') \\ &= \emptyset \wedge \mathfrak{J}(\mathbb{T}') \\ &= \emptyset \end{aligned}$$

したがって，補題 1.13 より $\mathbb{T}'\mathbb{T}$ は $\emptyset \vee x$ 可積分である．よって，cain の性質 (ii) より

$$\begin{aligned} \int \mathbb{T}' d\emptyset \int \mathbb{T} dx &= \int \mathbb{T}' \mathbb{T} d(\emptyset \vee x) \\ &= \int \mathbb{T}' \mathbb{T} dx \\ \therefore \mathbb{T}' \int \mathbb{T} dx &= \int \mathbb{T}' \mathbb{T} dx \end{aligned}$$

□

補題 1.17 $\forall x, y \in \mathbb{L}$,

$$x \wedge y = \emptyset \Leftrightarrow y \leq \bar{x}$$

証明 $(\Rightarrow)$ を示す．

$$\begin{aligned} \bar{x} &= \emptyset \vee \bar{x} \\ &= (x \wedge y) \vee \bar{x} \\ &= (x \vee \bar{x}) \wedge (y \vee \bar{x}) \\ &= y \vee \bar{x} \\ \therefore y &\leq \bar{x} \end{aligned}$$

($\Leftarrow$) を示す．

$$\begin{aligned} x \wedge y &= \bar{\bar{x}} \wedge \bar{\bar{y}} \\ &= \overline{(\bar{x} \vee \bar{y})} \\ &= \overline{((\bar{x} \vee y) \vee \bar{y})} \ (\because y \leq \bar{x}) \\ &= \overline{(\bar{x} \vee (y \vee \bar{y}))} \\ &= \overline{(\bar{x} \vee \top)} \\ &= \overline{\top} \\ &= \emptyset \end{aligned}$$

□

定理 1.18 $\forall x, y, z \in \mathbb{L}$,

$$x > z,\ y \wedge z = \emptyset \Rightarrow \int \mathbb{T}_y^x dz = \mathbb{T}_y^{x \wedge \bar{z}}$$

証明　$x > z$ より $x \neq \emptyset$ であるので $\mathbb{T}_y^x = \mathbb{T}^{x \vee y} \mathbb{T}_y$ が成り立つ．よって，

$$\begin{aligned} \int \mathbb{T}_y^x dz &= \int \mathbb{T}^{x \vee y} \mathbb{T}_y dz \\ &= \mathbb{T}_y \int \mathbb{T}^{x \vee y} dz \quad (\because \text{定理 1.16}) \\ &= \mathbb{T}_y \mathbb{T}^{(x \vee y) \wedge \bar{z}} \end{aligned}$$

ここで，$y \wedge z = \emptyset$ であるから補題 1.17 より $y \leq \bar{z}$ が言える．したがって，

$$\begin{aligned} (x \vee y) \wedge \bar{z} &= (x \wedge \bar{z}) \vee (y \wedge \bar{z}) \\ &= (x \wedge \bar{z}) \vee y \\ \therefore \mathbb{T}_y \mathbb{T}^{(x \vee y) \wedge \bar{z}} &= \mathbb{T}_y \mathbb{T}^{(x \wedge \bar{z}) \vee y} \end{aligned}$$

いま，$x > z$ より $x \wedge \bar{z} = x \neq \emptyset$ であるから $\mathbb{T}_y \mathbb{T}^{(x \wedge \bar{z}) \vee y} = \mathbb{T}_y^{x \wedge \bar{z}}$ が成り立つ．よって以上より，

$$\int \mathbb{T}_y^x dz = \mathbb{T}_y^{x \wedge \bar{z}} \tag{1.23}$$

□

定理 1.19 $\mathbb{L}$ の任意の元 x, y に対して以下が成り立つ．

(i)　$\int \mathbb{T}^x dx = 1$

(ii)　$x \neq \emptyset$ ならば $\int \mathbb{T}_y^x d(x \wedge \bar{y}) = 1$

証明　(i) を示す．cain の性質より，

$$\begin{aligned}\int \mathbb{T}^x dx &= \mathbb{T}^{x\wedge\bar{x}} \\ &= \mathbb{T}^{\emptyset} \\ &= 1\end{aligned}$$

(ii) を示す．$x \neq \emptyset$ より $\mathbb{T}^x_y = \mathbb{T}^{x\vee y}\mathbb{T}_y$ であり，また

$$\begin{aligned}x \vee y &= (x \vee y) \wedge \top \\ &= (x \vee y) \wedge (\bar{y} \vee y) \\ &= (x \wedge \bar{y}) \vee y\end{aligned}$$

よって，$\mathbb{T}^{x\vee y} = \mathbb{T}^{(x\wedge\bar{y})\vee y}$ は $(x \wedge \bar{y})$ 可積分であり，$\mathbb{T}_y$ は $(x \wedge \bar{y})$ 可積分でない．ゆえに，

$$\begin{aligned}\int \mathbb{T}^x_y d(x \wedge \bar{y}) &= \int \mathbb{T}^{x\vee y}\mathbb{T}_y d(x \wedge \bar{y}) \\ &= \mathbb{T}_y \int \mathbb{T}^{x\vee y} d(x \wedge \bar{y}) \quad (\because \text{定理 1.16}) \\ &= \mathbb{T}_y \mathbb{T}^{(x\vee y)\wedge\overline{(x\wedge\bar{y})}} \\ &= \mathbb{T}_y \mathbb{T}^{(x\vee y)\wedge(\bar{x}\vee y)} \\ &= \mathbb{T}_y \mathbb{T}^{(x\wedge\bar{x})\vee y} \\ &= \mathbb{T}_y \mathbb{T}^{y} \\ &= 1\end{aligned}$$

□

上の定理は

$$\int_{-\infty}^{\infty} f(x)dx = 1, \int_{-\infty}^{\infty} f(x|y)dx = 1$$

という確率密度関数と条件付き確率密度関数が満たすべき性質を表している．

以下，$\int$ に関する書き換えを簡単にする系を述べる．

系 1.20 任意の $\mathbb{T}'$ に対して，以下が成り立つ．

(i) $\mathfrak{J}(\mathbb{T}') \wedge y = \emptyset$ ならば

(1.24)
$$\int \mathbb{T}'\mathbb{T}^y dy = \mathbb{T}'$$

(ii) $\mathfrak{J}(\mathbb{T}') \wedge z = \emptyset$ かつ $y \wedge z = \emptyset$ ならば

(1.25)
$$\int \mathbb{T}'\mathbb{T}^{y\vee z} dz = \mathbb{T}'\mathbb{T}^y$$

(iii) $x \wedge z = \emptyset$ かつ $y \wedge z = \emptyset$ かつ $\Im(\Pi') \wedge z = \emptyset$ ならば

$$\int \Pi' \Pi^{x \vee y \vee z} dz = \Pi' \Pi^{x \vee y} \tag{1.26}$$

(iv) $x > \emptyset$ かつ $x \wedge y = y \wedge z = \emptyset$ のとき，

$$\int \Pi_z^{x \vee y} dy = \Pi_z^x \tag{1.27}$$

証明

(i) 定理 1.16 より，

$$\begin{aligned} \int \Pi' \Pi^y dy &= \Pi' \int \Pi^y dy \\ &= \Pi' \cdot 1 \\ &= \Pi' \end{aligned}$$

(ii) 定理 1.16 より，

$$\begin{aligned} \int \Pi' \Pi^{y \vee z} dz &= \Pi' \int \Pi^{y \vee z} dz \\ &= \Pi' \Pi^{(y \vee z) \wedge \bar{z}} \\ &= \Pi' \Pi^{(y \wedge \bar{z}) \vee (z \wedge \bar{z})} \\ &= \Pi' \Pi^{y \wedge \bar{z}} \\ &= \Pi' \Pi^y \ (\because \text{定理 1.17}) \end{aligned}$$

(iii) 定理 1.16 より，

$$\begin{aligned} \int \Pi' \Pi^{x \vee y \vee z} dz &= \Pi' \int \Pi^{x \vee y \vee z} dz \\ &= \Pi' \Pi^{(x \vee y \vee z) \wedge \bar{z}} \\ &= \Pi' \Pi^{\{(x \vee y) \wedge \bar{z}\} \vee (z \wedge \bar{z})} \\ &= \Pi' \Pi^{(x \vee y) \wedge \bar{z}} \end{aligned}$$

ここで，$x \wedge z = y \wedge z = \emptyset$ から $(x \vee y) \wedge z = \emptyset$ であるので定理 1.17 より

$$\Pi' \Pi^{(x \vee y) \wedge \bar{z}} = \Pi' \Pi^{x \vee y}$$

(iv)

$$\begin{aligned} \int \Pi_z^{x \vee y} dy &= \int \Pi^{x \vee y \vee z} \Pi_z dy \ (\because x > \emptyset) \\ &= \Pi_z \int \Pi^{x \vee y \vee z} dy \ (\because \text{定理 1.16}) \end{aligned}$$

$$
\begin{aligned}
&= \mathbb{T}_z \mathbb{T}^{(x\vee y\vee z)\wedge\bar{y}} \\
&= \mathbb{T}_z \mathbb{T}^{(x\vee z)\wedge\bar{y}} \\
&= \mathbb{T}_z \mathbb{T}^{x\vee z} \ (\because \text{定理 1.17}) \\
&= \mathbb{T}^x_z
\end{aligned}
$$

□

定理 1.21 $x \wedge y = \emptyset$ を満たす任意の $x, y, z \in \mathbb{L}$ に対して，以下が成り立つ．

(i) $\mathfrak{J}(\tilde{\mathbb{T}}) \leq \bar{y}$ かつ $\mathfrak{J}(\hat{\mathbb{T}}) \leq \bar{x}$ であるような $\tilde{\mathbb{T}}, \hat{\mathbb{T}}$ に対して，

$$\mathbb{T}^{x\vee y\vee z} = \tilde{\mathbb{T}}\hat{\mathbb{T}} \Rightarrow \mathbb{T}^{x\vee y\vee z} = \mathbb{T}^{(x\vee z)\wedge\bar{y}} \mathbb{T}^{(y\vee z)\wedge\bar{x}} \mathbb{T}_{z\wedge\bar{x}\wedge\bar{y}}$$

(ii) $x, y, z(> \emptyset)$ が互いに排反であるとき (i) の $\tilde{\mathbb{T}}\hat{\mathbb{T}}$ に対して，

$$\mathbb{T}^{x\vee y\vee z} = \tilde{\mathbb{T}}\hat{\mathbb{T}} \Rightarrow \mathbb{T}^{x\vee y\vee z} = \mathbb{T}^{(x\vee z)} \mathbb{T}^{(y\vee z)} \mathbb{T}_z$$

定義 1.22 (条件付き独立) $x, y, z \in \mathbb{L}$ に対して，

$$\mathbb{T}^x_{y\vee z} = \mathbb{T}^x_z \tag{1.28}$$

が成り立つとき，x は z を与えられた下で y と独立であるという．このとき，$x \perp\!\!\!\perp y \mid z$ と書く．$z = \emptyset$ のとき，$x \perp\!\!\!\perp y$ とかく．

定理 1.23 $x > \emptyset$ であるとき，

$$x \perp\!\!\!\perp y \mid z \Leftrightarrow \mathbb{T}^{x\vee y\vee z} = \mathbb{T}^{y\vee z}\mathbb{T}^x_z$$

証明

$$
\begin{aligned}
x \perp\!\!\!\perp y \mid z &\Leftrightarrow \mathbb{T}^x_{y\vee z} = \mathbb{T}^x_z \\
&\Leftrightarrow \mathbb{T}^{x\vee y\vee z}\mathbb{T}_{y\vee z} = \mathbb{T}^x_z \\
&\Leftrightarrow \mathbb{T}^{x\vee y\vee z} = \mathbb{T}^{y\vee z}\mathbb{T}^x_z
\end{aligned}
$$

□

定理 1.24 (因数分解定理) $x, y, z(> \emptyset) \in \mathbb{L}$ は互いに排反であるとき，

$$x \perp\!\!\!\perp y \mid z \Leftrightarrow \exists\, \tilde{\mathbb{T}}, \hat{\mathbb{T}}\ s.t.\ \mathfrak{J}(\tilde{\mathbb{T}}) \leq \bar{x},\ \mathfrak{J}(\hat{\mathbb{T}}) \leq \bar{y},\ \mathbb{T}^{x\vee y\vee z} = \tilde{\mathbb{T}}\hat{\mathbb{T}}$$

証明 $(\Rightarrow)$ を示す．定理 1.23 より，

$$\mathbb{T}^{x\vee y\vee z} = \mathbb{T}^{y\vee z}\mathbb{T}^x_y$$

が成り立つ．このとき，

$$\begin{aligned} x \wedge \mathfrak{J}(\mathbb{T}^{y\vee z}) &= x \wedge (y \vee z) \\ &= (x \wedge y) \vee (x \wedge z) \\ &= \emptyset \vee \emptyset \\ &= \emptyset \end{aligned}$$

よって，補題 1.17 より $\mathfrak{J}(\mathbb{T}^{y\vee z}) \leq \bar{x}$ が成り立つ．

(i) $z \geq x$ のとき，$x \vee z = z$ であるから，

$$\begin{aligned} \mathbb{T}^x_z = \mathbb{T}^{x\vee z}\mathbb{T}_z = \mathbb{T}^z\mathbb{T}_z = 1 \\ \therefore y \wedge \mathfrak{J}(\mathbb{T}^x_z) = y \wedge \mathfrak{J}(1) = y \wedge \emptyset = \emptyset \\ \therefore \mathfrak{J}(\mathbb{T}^x_z) \leq \bar{y} \end{aligned}$$

(ii) $z \geq x$ でないとき，$\mathbb{T}^x_z = \mathbb{T}^{x\vee z}\mathbb{T}_z = \mathbb{T}^{x\vee z}(\mathbb{T}^z)^{-1}$ であるので context の定義より，$\mathfrak{J}(\mathbb{T}^x_z) = x \vee z$ である．よって，

$$\begin{aligned} y \wedge \mathfrak{J}(\mathbb{T}^x_z) &= y \wedge (x \vee z) \\ &= (y \wedge x) \vee (y \wedge z) \\ &= \emptyset \vee \emptyset \\ &= \emptyset \\ \therefore \mathfrak{J}(\mathbb{T}^x_z) &\leq \bar{y} \end{aligned}$$

(i) (ii) より $\mathfrak{J}(\mathbb{T}^x_z) \leq \bar{y}$ が成り立つので，$\tilde{\mathbb{T}} = \mathbb{T}^{y\vee z}, \hat{\mathbb{T}} = \mathbb{T}^x_z$ とすればよい．

(⇐) を示す．

$$\begin{aligned} \mathbb{T}^{y\vee z}\mathbb{T}^x_z &= \mathbb{T}^{y\vee z}\mathbb{T}^{x\vee z}\mathbb{T}_z \\ &= \mathbb{T}^{x\vee z}\mathbb{T}^{y\vee z}\mathbb{T}_z \end{aligned}$$

ここで，補題 1.17 より

$$\begin{aligned} \mathfrak{J}(\tilde{\mathbb{T}}) \leq \bar{x} \Leftrightarrow x \wedge \mathfrak{J}(\tilde{\mathbb{T}}) = \emptyset \\ \mathfrak{J}(\hat{\mathbb{T}}) \leq \bar{y} \Leftrightarrow y \wedge \mathfrak{J}(\hat{\mathbb{T}}) = \emptyset \end{aligned}$$

であり，$x \wedge y = \emptyset, \mathbb{T}^{x\vee y\vee z} = \tilde{\mathbb{T}}\hat{\mathbb{T}}$ であるので定理 1.21 より

$$\begin{aligned} \mathbb{T}^{x\vee z}\mathbb{T}^{y\vee z}\mathbb{T}_z &= \mathbb{T}^{x\vee y\vee z} \\ \therefore \mathbb{T}^{x\vee y\vee z} &= \mathbb{T}^{y\vee z}\mathbb{T}^x_z \end{aligned}$$

よって，定理 1.23 より $x \perp\!\!\!\perp y \mid z$ である．

□

1.6　M-Law と N-Law

この先の議論を展開するにあたって重要な役割を果たす定理を 2 つ紹介する

定理 1.25 (M-Law) $x \wedge y = \emptyset$ とする．$x \wedge a > \emptyset$，$y \wedge b > \emptyset$ を満たす任意の $a, b \in \mathbb{L}$ に対して

$$x \perp\!\!\!\perp y | z \Rightarrow (x \wedge a) \perp\!\!\!\perp (y \wedge b) | z \tag{1.29}$$

が成立し，特に $z = \emptyset$ のとき $x \perp\!\!\!\perp y \Rightarrow (x \wedge a) \perp\!\!\!\perp (y \wedge b)$

証明　$x = \emptyset$ または $y = \emptyset$ のときは仮定を満たさないため $x \neq \emptyset$ かつ $y \neq \emptyset$ と出来る．
仮定と定理 1.23 より $\Pi_z^{x \vee y} = \Pi_z^x \Pi_z^y$．$x \wedge y = \emptyset$ なので，

$$\begin{aligned}
\int \Pi_z^{x \vee y} d(y \wedge \bar{b} \wedge \bar{z}) &= \Pi_z^{(x \vee y) \wedge \overline{y \wedge \bar{b} \wedge \bar{z}}} \\
&= \Pi_z^{(x \vee y) \wedge (\bar{y} \vee b \vee z)} \\
&= \Pi_z^{(x \vee y) \wedge (\bar{y} \vee b)} \\
&= \Pi_z^{x \vee (y \wedge b)}
\end{aligned}$$

同じように $(x \vee z) \wedge (y \wedge \bar{b} \wedge \bar{z}) = \emptyset$ なので，

$$\begin{aligned}
\int \Pi_z^x \cdot \Pi_z^y d(y \wedge \bar{b} \wedge \bar{z}) &= \Pi_z^x \cdot \int \Pi_z^y d(y \wedge \bar{b} \wedge \bar{z}) \\
&= \Pi_z^x \cdot \Pi_z^{y \wedge \overline{y \wedge \bar{b} \wedge \bar{z}}} \\
&= \Pi_z^x \cdot \Pi_z^{y \wedge (b \vee z)} \\
&= \Pi_z^x \cdot \Pi_z^{y \wedge b}
\end{aligned}$$

以上 3 つの等式より

$$\Pi_z^{x \vee (y \wedge b)} = \Pi_z^x \cdot \Pi_z^{y \wedge b}$$

上と同じような議論を繰り返すことで，

$$\int \Pi_z^{x \vee (y \wedge b)} d(x \wedge \bar{a} \wedge \bar{z}) = \Pi_z^{(x \wedge a) \vee (y \wedge b)}$$

$$\int \Pi_z^x \cdot \Pi_z^{(y \wedge b)} d(x \wedge \bar{a} \wedge \bar{z}) = \Pi_z^{(x \wedge a)} \cdot \Pi_z^{y \wedge b}$$

となり，

$$\Pi_z^{(x \wedge a) \vee (y \wedge b)} = \Pi_z^{(x \wedge a)} \cdot \Pi_z^{y \wedge b}$$

以上より題意は示された．　□

定理 1.26 (N-Law) $\Im(\top^x_{y\vee z}) \le \bar{z}$ 且つ $x \wedge \bar{z} > \emptyset$ ならば

$$\top^x_{y\vee z} = \top^{x\wedge\bar{z}}_{y\wedge\bar{z}} \qquad (x > \emptyset) \tag{1.30}$$

証明　$x \le y \vee z$ のとき，$x \wedge \bar{z} \le (y \vee z) \wedge \bar{z} \Leftrightarrow x \wedge \bar{z} \le y \wedge \bar{z}$ となるため，命題 1 より両辺ともに 1 となり成立.

$\neg x \le y \vee z$ のとき，$\Im(\top^x_{y\vee z}) = x \vee y \vee z$. よって $x \vee y \vee z \le \bar{z}$.

$$\begin{aligned}
\top^x_{y\vee z} &= \top^x_{y\vee z} \cdot \top^{y\wedge\bar{z}} \cdot \top_{y\wedge\bar{z}} \\
&= \top^x_{y\vee z} \cdot \top^{(y\vee z)\wedge\bar{z}} \cdot \top_{y\wedge\bar{z}} \\
&= \top^x_{y\vee z} \cdot \int \top^{y\vee z} dz \cdot \top_{y\wedge\bar{z}} \\
&= \int \top^x_{y\vee z} \cdot \top^{y\vee z} dz \cdot \top_{y\wedge\bar{z}} \\
&= \int \top^{x\vee y\vee z} dz \cdot \top_{y\wedge\bar{z}} \\
&= \top^{(x\vee y\vee z)\wedge\bar{z}} \cdot \top_{y\wedge\bar{z}} \\
&= \top^{(x\wedge\bar{z})\vee(y\wedge\bar{z})} \cdot \top_{y\wedge\bar{z}} \\
&= \top^{x\wedge\bar{z}}_{y\wedge\bar{z}}
\end{aligned}$$

以上より成立. □

1.7 Graphoid と Separoid

この節では条件付き独立が満たすべき性質として一般的に知られている Pearl & Paz（1987）によって提案された Graphoid と Dawid（2001）によって提案された Separoid について述べ，Cain がこれらの性質を満たしていることを確かめる.

定義 1.27 ブール束 $\mathbb{L}$ 上で定義された三項関係 $\cdot \amalg \cdot | \cdot$ が以下の性質を満たすとき $\mathbb{L}$ を graphoid という．ただし，すべての要素は $\emptyset$ でなく，かつ互いに排反である.

G1 :	$x \amalg y \mid z \;\Rightarrow\; y \amalg x \mid z$	(Symmetry)
G2 :	$x \amalg (y \vee w) \mid z \;\Rightarrow\; x \amalg y \mid z$	(Decomposition)
G3 :	$x \amalg (y \vee z) \mid w \;\Rightarrow\; x \amalg y \mid (z \vee w)$	(Weak union)
G4 :	$x \amalg y \mid (z \vee w),\ x \amalg z \mid w \;\Rightarrow\; x \amalg (y \vee z) \mid w$	(Contraction)
G5 :	$x \amalg y \mid (z \vee w),\ x \amalg z \mid (y \vee w) \;\Rightarrow\; x \amalg (y \vee z) \mid w$	(Intersection) .

定理 1.28 $\cdot \perp\!\!\!\perp \cdot | \cdot$ は graphoid の性質を満たす.

定義 1.29 (separoid) $(S, \leq)$ を join-semilattice(結び半束) とする．$\cdot \perp \cdot | \cdot$ を S 上で定義された三項関係とし，以下の性質を満たすとき $(S, \leq, \perp)$ は *separoid* であるという．

$$
\begin{array}{lll}
\text{P1:} & x \perp y \,|\, x & \\
\text{P2:} & x \perp y \,|\, z & \Rightarrow\ y \perp x \,|\, z \\
\text{P3:} & x \perp y \,|\, z \ \&\ w \leq y & \Rightarrow\ x \perp w \,|\, z \\
\text{P4:} & x \perp y \,|\, z \ \&\ w \leq y & \Rightarrow\ x \perp y \,|\, (z \vee w) \\
\text{P5:} & x \perp y \,|\, z \ \&\ x \perp w \,|\, (y \vee z) & \Rightarrow\ x \perp (y \vee w) \,|\, z .
\end{array}
$$

ここで，$(S, \leq)$ が lattice であり，さらに以下の P6 を満たすとき，separoid $(S, \leq, \perp)$ は *strong separoid* であるという．

$$
\begin{array}{ll}
\text{P6:} & \text{If } z \leq y \ \&\ w \leq y \text{ then} \\
& x \perp y \,|\, z \ \&\ x \perp y \,|\, w \quad \Longrightarrow\ x \perp y \,|\, (z \wedge w) .
\end{array}
$$

定理 1.30 $(\mathbb{L}, \leq, \perp\!\!\!\perp)$ は strong separoid である．

定理 1.28,1.30 が成り立つことをいうために，$\cdot \perp\!\!\!\perp \cdot | \cdot$ についてのいくつかの定理を紹介しよう．

定理 1.31 (symmetry) $x, y, z \neq \emptyset$ であるとする．$x \perp\!\!\!\perp y|z \Leftrightarrow y \perp\!\!\!\perp x|z$

証明　定理 1.23 より

$$
\begin{aligned}
x \perp\!\!\!\perp y|z &\Leftrightarrow \mathbb{T}^x_{y \vee z} = \mathbb{T}^x_z \\
&\Leftrightarrow \mathbb{T}^{x \vee y}_z = \mathbb{T}^x_z \mathbb{T}^y_z \\
&\Leftrightarrow \mathbb{T}^{y \vee x}_z = \mathbb{T}^y_z \mathbb{T}^x_z \\
&\Leftrightarrow y \perp\!\!\!\perp x|z
\end{aligned}
$$

□

定理 1.32 (decomposition) $x \wedge y = \emptyset$ ならば，任意の $a \leq x, b \leq y$ に対して，

$$x \perp\!\!\!\perp y|z \Rightarrow a \perp\!\!\!\perp b|z$$

M-law より明らかであるため証明は省略する．

定理 1.33 (weak union) x, y, z, w はすべて $\emptyset$ でなく, それぞれ互いに排反であるとすると $x \perp\!\!\!\perp (y \vee z)|w \ \Rightarrow x \perp\!\!\!\perp y|(z \vee w)$

証明　$x \perp\!\!\!\perp (y \vee z)|w$ より $\mathbb{T}^{x \vee y \vee z \vee w} = \mathbb{T}^x_w \mathbb{T}^{y \vee z \vee w}$．M-law より，$\mathbb{T}^x_{z \vee w} = \mathbb{T}^x_w$．これを合わせて $\mathbb{T}^{x \vee y \vee z \vee w} = \mathbb{T}^x_{z \vee w} \mathbb{T}^{y \vee z \vee w}$．これより $x \perp\!\!\!\perp y|(z \vee w)$　□

定理 1.34 (contraction) x, y, z, w はすべて $\emptyset$ でなく, それぞれ互いに排反であるとすると, $(x \perp\!\!\!\perp y \mid (z \vee w)$ かつ $x \perp\!\!\!\perp z \mid w) \Leftrightarrow x \perp\!\!\!\perp (y \vee z) \mid w$

証明

($\Leftarrow$) $x \perp\!\!\!\perp (y \vee z) | w$ に M-law を作用させて $x \perp\!\!\!\perp z | w$, これより

$$\mathbb{T}^{x \vee y \vee z \vee w} = \mathbb{T}^{x}_{w} \mathbb{T}^{y \vee z \vee w} = \mathbb{T}^{x}_{z \vee w} \mathbb{T}^{y \vee z \vee w}$$

以上より $x \perp\!\!\!\perp y \mid (z \vee w)$

($\Rightarrow$) $x \perp\!\!\!\perp y | (z \vee w)$ かつ $x \perp\!\!\!\perp z | w$ より $\mathbb{T}^{x \vee y \vee z \vee w} = \mathbb{T}^{x \vee z \vee w} \mathbb{T}^{y}_{z \vee w}$ かつ $\mathbb{T}^{x \vee z \vee w} = \mathbb{T}^{x \vee w} \mathbb{T}^{z}_{w}$. よって $\mathbb{T}^{x \vee y \vee z \vee w} = \mathbb{T}^{x \vee w} \mathbb{T}^{z}_{w} \mathbb{T}^{y}_{z \vee w}$ となり, M-law より $\mathbb{T}^{y \vee z \vee w} = \mathbb{T}^{w} \mathbb{T}^{z}_{w} \mathbb{T}^{y}_{z \vee w} \Leftrightarrow \mathbb{T}^{y \vee z}_{w} = \mathbb{T}^{z}_{w} \mathbb{T}^{y}_{z \vee w}$. これより $\mathbb{T}^{x \vee y \vee z \vee w} = \mathbb{T}^{x \vee w} \mathbb{T}^{y \vee z}_{w}$ となり $x \perp\!\!\!\perp (y \vee z) | w$ を得る.

□

定理 1.35 (intersection) x, y, z, w はすべて $\emptyset$ でなく, それぞれ互いに排反であるとすると, $(x \perp\!\!\!\perp y \mid (z \vee w)$ かつ $x \perp\!\!\!\perp z \mid (y \vee w)) \Leftrightarrow x \perp\!\!\!\perp (y \vee z) \mid w$.

証明

($\Leftarrow$) 定理 1.23 より $x \perp\!\!\!\perp y | (z \vee w) \Rightarrow \mathbb{T}^{x \vee y \vee z \vee w} = \mathbb{T}^{x}_{z \vee w} \mathbb{T}^{y \vee z \vee w}$ かつ $x \perp\!\!\!\perp z | (y \vee w) \Rightarrow \mathbb{T}^{x \vee y \vee z \vee w} = \mathbb{T}^{x}_{y \vee w} \mathbb{T}^{y \vee z \vee w}$. 二つを合わせて, $\mathbb{T}^{x}_{z \vee w} = \mathbb{T}^{x}_{y \vee w}$ となる. N-law より, $\mathbb{T}^{x}_{z \vee w} = \mathbb{T}^{x}_{w}$, よって $\mathbb{T}^{x \vee y \vee z \vee w} = \mathbb{T}^{x}_{w} \mathbb{T}^{y \vee z \vee w} \Leftrightarrow x \perp\!\!\!\perp (y \vee z) | w$.

($\Rightarrow$) $x \perp\!\!\!\perp (y \vee z) | w$ なので M-law より $(x \perp\!\!\!\perp y | w$ かつ $x \perp\!\!\!\perp z | w) \Leftrightarrow (\mathbb{T}^{x}_{y \vee w} = \mathbb{T}^{x}_{w}$ かつ $\mathbb{T}^{x}_{z \vee w} = \mathbb{T}^{x}_{w})$. さらに, $x \perp\!\!\!\perp (y \vee z) | w \Leftrightarrow \mathbb{T}^{x \vee y \vee z \vee w} = \mathbb{T}^{x}_{w} \mathbb{T}^{y \vee z \vee w}$ より, $\mathbb{T}^{x \vee y \vee z \vee w} = \mathbb{T}^{x}_{y \vee w} \mathbb{T}^{y \vee z \vee w}$ and $\mathbb{T}^{x \vee y \vee z \vee w} = \mathbb{T}^{x}_{z \vee w} \mathbb{T}^{y \vee z \vee w}$. 以上より $x \perp\!\!\!\perp z | (y \vee w)$ かつ $x \perp\!\!\!\perp y | (z \vee w)$.

□

定理 1.31~1.35 より, $\cdot \perp\!\!\!\perp \cdot | \cdot$ が G1~G5 を満たすことが言えたため, 定理 1.28 については成立することが言えた. 次に P1~P6 についても成立することを見ていこう. なお, P1~P5 については join-semilattice を考えているため要素が $\emptyset$ となる場合については考えず, またどの要素も互いに排反であるとする.

証明

P1: $x \leq x$ なので, 命題 4 より $x \perp\!\!\!\perp y | x$.
P2: 定理 14 より成立.

P3: 定理 15 より成立.

P4: 仮定より $x \perp\!\!\!\perp (y \vee w)|z$ となるため，定理 16 より成立.

P5: 定理 17 より成立.

P6: 仮定より $\Pi^x_{y\vee z} = \Pi^x_z = \Pi^x_y$ かつ $\Pi^x_{y\vee w} = \Pi^x_w = \Pi^x_y$
よって $\Pi^x_z = \Pi^x_w$ となるため，定理 5 より $\Pi^x_z = \Pi^x_{z\wedge w}$ が成立する．これと仮定より $\Pi^x_{y\vee(z\wedge w)} = \Pi^x_{(y\vee z)\wedge(y\vee w)} = \Pi^x_y = \Pi^x_{z\wedge w} \Leftrightarrow x \perp\!\!\!\perp y|(z \wedge w)$ を得る.

以上より定理 1.30 についても成立することが言えた. □

2. Coq/SSReflect

2.1 形式化と証明支援系

形式化とは，数学の証明において用いた公理や規則などを，漏れなくすべて記述することを意味する．形式化をすることの意味は，議論や論理の曖昧さを完全に排除できるという点にあるだけでなく，コンピューターの言語として記述できるという点にある．人間が，ある定理の証明の正しさを検証するには，時間と人数をかけるしかないが，それをどんなにかけたとしても必ずどこかに曖昧さが残ってしまう．それをコンピューターにやらせてしまえば，コンピューター自身に不備がない限り，曖昧さのない完璧な検証が出来るであろう．また，コンピューターが検証できるのならば，今度はある定理の証明を自動で行ってくれるという，自動証明に期待がかかってくる．近年の機械学習や人工知能の分野の発展から考えると，そう遠くない未来に実現できそうである．そのために我々が今できることは，様々な定理の証明をコンピューターで検証しておき，ライブラリという形で後世に可能な限り残しておく，ということである.

ここで，先に述べたコンピューターの言語として記述する際に使うのが，証明支援系である．証明支援系の例としては HOL，Isabelle，Coq 等がある．証明支援系を用いた証明の業績で有名なものは，2004 年の Georges Gonthier による四色定理の形式化（Coq/SSReflect）と，2014 年 8 月の Hales らのプロジェクトチームによるケプラーの予想の証明の形式化（HOL Light，一部 SSReflect）である.

本論文で行った形式化は，Coq とその拡張である SSReflect を用いた．次節はこれらについて簡単に紹介を行う.

2.2 Coq/SSReflect

定理証明支援系 Coq [2] は，フランス INRIA（フランス国立情報学自動制御研究所）で開発された今最も使われている証明支援系である．SSReflect (small scale reflection) は，Coq の拡張であり，Coq だけ用いるよりもタクティックの数が大幅に減るだけでなく，よ

り紙とペンで書いた証明に近いコードを作ることができる．また，mathcomp [3] と呼ばれるライブラリも存在し，これには有限集合や，$\sum$ や $\prod$ などの同じ演算子を何度も使った計算の一般的に成り立つ性質などが含まれている．これを含めて SSReflect と呼ぶことが多い．`http://math-comp.github.io/math-comp/` にて，windows 向けに提供されている mathcomp のインストーラーを用いると自動的に SSReflect のライブラリが含まれた形でインストールされる．本論文で用いた Coq のバージョンは Coq 8.5 で，SSReflect や MathComp のバージョンは 1.6 である．

この節で簡単に証明の仕方などを紹介するが，より具体的なタクティックの使い方は，”Program and Proofs” [4], “定理証明支援系 Coq による形式検証” [5], “An Introduction to small scale reflection in Coq” [6], “A small scale reflection extension for the Coq system.” [7] を参照されたい．

一例として，自然数は以下のように帰納的に定義されている．

```
Inductive nat : Set :=  O : nat | S : nat → nat
```

これは

- `O` が `nat` 型である．
- `S` は `nat` 型を受け取り `nat` 型を返す関数で，これ自身も `nat` 型である．

ということを意味している．この定義により，`O` を自然数 0 とみて，`S(O)` を 1，`S(S(O))` を 2... のようにして考えれば，帰納的にすべての自然数が定義されたことになる．

またこの論文でよく使われる型や定義についてもいくつか紹介しておく．

`eqType`，`choiceType` 型

等しさが決定可能な型が `eqType` である．ここでの等しさとは Leibniz equality のことを指している．これを用いると SSRefrect のライブラリにある便利な補題が使える．主に Lattice を形式化するのに用いた．また，`choiceType` はこの `eqType` をもとに作られた型で，`eqType` は任意の型に対して様々な補題が定義されているのに対し，`choiceType` は `eqType` に対して同様の補題が定義されていると考えて良い．

`int` 型

`int` 型はライブラリ `ssrint` で定義されている，いわゆる整数のようなものを定義している．

```
CoInductive int : Set := Posz of nat | Negz of nat.
```

とても簡単な解説をすれば，`Posz` は正の整数，`Negz` は負の整数のことである．主に coin を raising coin の積の形で表した時の肩の数を表すのに使われている．

seq 型

これは何かを並べた列を表す型である．

```
Notation seq := list.
```

この list は Coq で定義されているもので，SSReflect で新しく seq と定義し直している．この元になっている list の定義を見てみると，

```
Inductive list (A : Type) : Type :=
    nil : list A | cons : A → list A → list A
```

これも nat 同様に帰納的に定義されているが，Type を引数に取ることができる．例えば nat を引数にとれば，自然数をつなげた列を定義することができる．nil は空集合を表していて，cons は，ある型の元と，その型で構成された列を引数に取り，新しく列を構成する関数で，これも list 型である．主に coin を構成するときに用いた．

big_op

これは，mathcomp 内の bigop.v で定義されていて，同じ演算を繰り返し用いるときに使う．自然数でいう $\sum$，$\prod$ もこの big を用いて定義されている．例えば，$\sum_{\substack{0\le i<n \\ P(i)}} F(i)$ は

```
\big[addn/0]_(0 ≤ i < n | P i) F i
\sum_(0 ≤ i < n | P i) F i
```

のように表すことができる．\sum の方は上の Notation による書き換えで SSReflect では主にこちらを用いている．自分で定義した演算に対しては，上の方法で表記するのが普通で [] 内には繰り返す演算と単位元が入る．() 内には繰り返しを用いる範囲の指定が入る．これも主に coin を形式化するときに用いた．

3. Cain Algebra の形式化

3.1 Lattice の形式化

まず，Lattice の形式化についての形式化を紹介する．これらのコードは “Formalization of probablistic conditional independence relation using Coq/SSreflect ” [8] を参考に作られている．まず，Lattice の条件を公理として認めさせる．

```
Parameter L : eqType.
Parameters bot top: L.
Parameters meet join : L → L → L.
```

```
Axiom join_commutative : commutative join.
Axiom join_associative : associative join.
Axiom join_absorption : ∀ x y, meet x (join x y) = x.
Axiom meet_commutative : commutative meet.
Axiom meet_associative : associative meet.
Axiom meet_absorption : ∀ x y, join x (meet x y) = x.
Axiom meet_distributive : ∀ x y z,
        meet x (join y z) = join (meet x y) (meet x z) .
```

まず，Lattice 自身の型は L と表記し，eqType とする．bot は ∅，top は ⊤ を表している．meet，join は二つの L 型の引数をとって，同じ L 型の元を返す型である．

Axiom で定義されてるのは，lattice の性質と分配則の性質を加えたものである．

また，Notation で書かれているものは join と meet をより，紙で書いた定義に近づけ見やすくするためだけのものである．

また，bot，top に関しては，L 型であることしか定義されていないが，順序関係を

```
Definition ge x y := join x y = x.
Definition gt x y := (ge x y) ∧ (x ≠ y).
Definition le x y := ge y x.
Definition lt x y := (le x y) ∧ (x ≠ y).
```

のように定義することで，これらに関する性質を以下のようにして定義することができる．

```
Axiom bot_minimum : ∀ x, ge x bot.
Axiom top_maximum : ∀ x, ge top x.
```

補元は以下のように定義した．

```
Parameter  com : L → L.
Axiom com_bot : ∀ x, meet x (com x)  = bot.
Axiom com_top : ∀ x, join x (com x)  = top.
```

これらについても複数の Lemma を証明してあるので詳しくはコードを参照していただきたい．

3.2 Cain Algebra の形式化

まずは Lattice の時同様，Cainoid で認める公理を設定する．

```
Parameter coins : choiceType.
Parameter dot : coins → coins → coins.
Parameter mix : L  → L → coins.
Definition bob := mix bot bot.
Definition up x := mix x bot.
Definition down x := mix bot x.
```

```
Axiom dotC : commutative dot.
Axiom dotA : associative dot.
Axiom bob_unitL : left_id bob dot.
Axiom up_down_unitL : ∀ x, dot (up x) (down x) = bob.
Axiom mix_up_down : ∀ x y,
        x ≠ bot → mix x y = dot (up (join x y)) (down y).
```

$\mathfrak{C}$ は coins と表記し，choiceType で定義した．mix は L の元を 2 つ取り coins を返す，すなわち $\mathbb{T}_y^x$ の役割を担っている．up，down はそれぞれ raising coin ，lowering coin であり，bob は bottom over bottom の略で単位元 1 に対応するものである．

これを用いて様々な定理を形式化していくが，形式化の特性上論理の飛躍を一切許さないので，1 つの定理の証明に必要な補題がいくつも出てくる．これらに関しては，証明部分を省き掲載するが，すべて証明が済んでいる．

```
Lemma  bob_unitR : right_id bob dot.
Lemma  down_up_unitL : ∀ x, dot (down x)  (up x) = bob.
Lemma  up_bot_bob : up bot = bob.
Lemma  down_bot_bob : down bot = bob.
```

定理 1.2 のベイズの定理は以下のように形式化できる．

```
Theorem Bayes_Theorem: ∀ x y, x ≠ bot  →  y ≠ bot  →
        mix x y =  dot  (mix y x) (dot (up x) (down y)).
Proof.
   move⇒ x y Hx Hy.
   rewrite [in RHS]mix_up_down;  last exact.
   by rewrite dotA -![dot (dot (up _) _) _]dotA
   [dot (_ x) (_ x)]dotC up_down_unitL bob_unitL
   join_commutative mix_up_down.
Qed.
```

逆元については，以下のように定義する．

```
Lemma dot_inv_sig: ∀ c:coins, {d | dot d c = bob}.
Definition dot_inv (c:coins) := sval (dot_inv_sig c).
Lemma dotV : left_inverse bob dot_inv dot.
```

まとめると，上の Lemma ですべての $\mathfrak{C}$ の元 c に対し，左から dot 演算で単位元になるような元が必ず存在することを示し，その掛けた元を dot_inv c と表記できるよう定義し，最後に下の段で dot_inv が左逆元であることを，Coq/SSReflect であらかじめ定義されている left_inverse を用いて証明し，その証明に dotV という名前を付けた，ということである．もとのライブラリ内で定義しているものに関連づけることは，ライブラリ内の大量の Lemma を用いることができるようになる，ということなので大変重要である．

補題 1.3 に関しては，Coq/SSReflect の Canonical を用いて，ライブラリ内のアーベル群の構造に埋め込むことができる．

```
Definition zmodMixin := ZmodMixin dotA dotC bob_unitL dotV.
Canonical zmodType := ZmodType _ zmodMixin.
```

これにより，

```
Notation x \+ y := (dot x y)(at level 50, left associativity).
Lemma dotrC:  ∀ a b,  a \+ b =  b \+ a.
Proof. by apply: GRing.addrC.  Qed.
```

のように Cainoid の可換律が，GRing.addrC という SSReflect 内の定理を用いて証明できる．

いよいよ $\mathfrak{C}$ の元の分解（定理 1.7）の形式化だが，その前にそれよりも条件の弱い補題 1.4「任意の Cainoid $\mathfrak{C}$ の元は raising coin のみの積の形で表記できる．」についての説明をする．以下が補題 1.4 に対応する定理である．

```
Theorem coin_prod_up : ∀ c : coins, {xns:seq (int*L) |
    c = \big[dot/bob]_(xn ← xns)  nupsz xn.1 xn.2}.
```

つまり，「任意の $\mathfrak{C}$ の元 c に対して，$c = \mathbb{T} = \left(\mathbb{T}^{x_1}\right)^{n_1} \cdots \left(\mathbb{T}^{x_r}\right)^{n_r}$ と書けるような，$(\mathbb{Z}, \mathbb{L})$ の列が存在する」ということである．

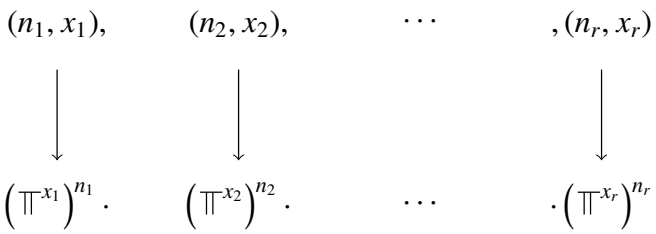

Fig. 1.　seq と coin の分解の対応

Fig.1 のように，$(\mathbb{Z}, \mathbb{L})$ の列に big_op をあてがうことで，coin の積が構成される．今後 coin の分解に関する操作はこのように $(\mathbb{Z}, \mathbb{L})$ の列を用いて行う．

さて，定理 1.7 だが，これと比較して補題 1.4 に足りない条件は，「各 x_i は互いに異なり，各 $n_i \neq 0$ である．」の部分である．これを追加すると以下のようなコードになる．

```
Theorem coin_prod_up_strong c: {xns:seq (int*L) |
    c = \big[dot/bob]_(xn ← xns)  nupsz xn.1 xn.2 ∧
    uniq ([seq i.2 | i ← xns]) ∧ 0 \notin [seq i.1 | i ← xns]}.
```

uniq ([seq i.2 | i ←xns]) の部分についてだが，これは，[seq i.2 | i ←xns] で，ペアの列の 2 つ目の要素を取り出した列を意味し，uniq で，引数の列に同じ要素がない

ことを保証している．xns の 2 つ目の要素は $\mathbb{L}$ であるので，「各 x_i は互いに異なり」の部分に相当する．0 \notin [seq i.1 | i ←xns] の部分は，[seq i.1 | i ←xns] で，ペアの列の 1 つ目の要素を取り出した列を意味し，\notin により，0 がその中に含まれない，つまり「各 $n_i \neq 0$ である．」の部分に相当する．

この証明をするのに以下の 2 つの関数を定義した．

```
Fixpoint head_in_tail (a:(int*L))(l:seq(int*L)):=
 match l with
 | nil ⇒ [::a]
 | h :: t ⇒ if a.2 = h.2 then ((a.1 + h.1),a.2)::t
  else h :: (head_in_tail a t)
end.

Fixpoint remove0L (xns:seq(int*L)):=
    match xns with
    | nil  ⇒ nil
    | h::t ⇒ if h.1 = 0 then remove0L t else h ::(remove0L t)
end.
```

この定理だけでなく，補題 1.4 等の証明は，「〜のような列が存在する．」というように定理を言い換えているため，数学的な証明と少しギャップがある．このような定理，補題を証明するのに，コード内では帰納法 elim を用いている．帰納法を用いることができる条件として，用いたいものの型が Inductive に定義されている必要があるが，Theorem 内で xns は seq で定義されていて，第 2.2 節より，これは Inductive で定義されている．

seq の elim による場合分けは，nil の場合と a :: l（すなわち cons a l）の場合に分かれるが，head_in_tail は a :: l によって外に出ている a の部分を l に組み込む際に用いている．

また，組み込んだものを coin の式に置き換えた時，必ず出てくるのが以下の Theorem である．

```
Theorem addnupsz n m x:
        dot (nupsz n x)(nupsz m x) = nupsz (n + m) x.
```

紙とペンで証明する際には，流してしまうような計算でも Coq は許してくれない．上の定理は $\left(\mathbb{T}^x\right)^n \cdot \left(\mathbb{T}^x\right)^m = \left(\mathbb{T}^x\right)^{n+m}$ と主張しているだけなのだが，肩の型が int であり，これも Inductive に定義され，さらに int 自体も nat という Inductive に定義されたものを使って定義されているため，場合分けが生じた．

また，remove0L は「各 $n_i \neq 0$ である．」の部分の証明に使われており，これは，xns の列の第 1 要素に 0 が含まれていたら，その元をすべて取り除く関数である．もちろん，肩が 0 である coin は単位元に等しいので取り除いても支障はない．これに関しては以下の定理で保障されている．

```
Theorem coin_prod_up_eq_non0 (xns:seq(int*L)):
    \big[dot/bob]_(xn ← xns)  nupsz xn.1 xn.2
    = \big[dot/bob]_(xn ← (remove0L xns))  nupsz xn.1 xn.2 .
```

これらの関数，補題，定理を用いて定理 1.7 を証明することができる．

次は，context の形式化である．

```
Parameter ctext : coins → L.
Axiom weak_ctext: ∀ (c:coins) (nx: seq(int*L)),
        c = \big[dot/bob]_(i ← nx)  nupsz i.1 i.2 →
        uniq [seq i.2 | i ← nx] → 0 \notin [seq i.1 | i ← nx] →
        ctext c = \big[join/bot]_(i ← nx)  i.2.
```

任意の coin に関して定理 1.7 の性質を満たすときに，$\mathfrak{J}$ が定義されるが，この前提となっている性質が，

```
        c = \big[dot/bob]_(i ← nx)  nupsz i.1 i.2 →
        uniq [seq i.2 | i ← nx] → 0 \notin [seq i.1 | i ← nx] →
```

で示されており，当然これは coin_prod_up_strong と同じ形をしている．ctext が $\mathfrak{J}$ にあたるもので，nx : seq(int*L) の 2 番目の要素をつなげたものとして定義している．今回は join を何回も用いてつなげるため，

```
        ctext c = \big[join/bot]_(i ← nx)  i.2.
```

のように定義した．さて，この context に関して，もっとも簡単な coin である atom coin に関する Lemma が成り立つ．

```
Lemma ctext1 x: ctext (up x) = x.
```

$$\mathfrak{J}(\mathbb{T}^x) = x$$

```
Lemma ctext2 x y : x ≠ bot ∧ ge y x → ctext (mix x y) = bot.
```

$x \neq \emptyset$ かつ $y \geq x$ ならば $\mathfrak{J}(\mathbb{T}^x_y) = \emptyset$

```
Lemma ctext3 x y :
        x ≠ bot ∧ ~~ ge y x → ctext (mix x y) = join x y.
```

$x \neq \emptyset$ かつ $y \geq x$ でないならば $\mathfrak{J}(\mathbb{T}^x_y) = x \vee y$

```
Lemma ctext4 x: ctext (down x) = x.
```

$$\mathfrak{J}(\mathbb{T}_x) = x$$

これは，cain に関する定理を証明するのに使うが，本論文では出てこないので紹介程度にとどめておく．以下は補題 1.10 に対応するコードである．

```
Theorem Th3_1 c : ctext c = ctext (dot_inv c).
```

これだけ見ると簡単に思えてしまうが，これは単に weak_ctext を用いるだけでは証明できないので，以下に用意した補題の例を挙げる．

```
Lemma dot_inv_lemma c1 c2:
        dot c1 c2 = bob → dot_inv c1 = c2.

Lemma dot_inv_big nx:
        dot (\big[dot/bob]_(i ← nx) nupsz i.1 i.2)
        (\big[dot/bob]_(i ← nx) nupsz (- i.1) i.2) = bob.

Lemma dot_inv_prod_up c nx:
        c = \big[dot/bob]_(i ← nx) nupsz i.1 i.2 →
        dot_inv c = \big[dot/bob]_(i ← nx) nupsz (- i.1) i.2 .

Fixpoint minusL (xns:seq(int*L)):=
  match xns with
  | nil ⇒ nil
  | h :: t ⇒ (-h.1, h.2)::(minusL t)
  end.
```

coin_prod_up_strong より c と dot_inv c には，定理 1.7 の性質を持つ seq (int * L) の元が存在するが，その第 1 要素の符号が反転したものなのかどうかはわからず，上のような Lemma がないと証明できないことがわかる．また，c に対する coin_prod_up_strong とこの 3 つの Lemma で，dot_inv c に関して

```
        dot_inv c = \big[dot/bob]_(i ← nx)  nupsz -i.1 i.2
        uniq [seq i.2 | i ← nx]
        0 \notin [seq i.1 | i ← nx]
```

が得られても，weak_ctext を使うことができない．1 つ目の条件と形が合わないからである（マイナスがついているだけでも Coq は認めてくれない）．従って，minus L が必要になる．これは引数 seq (int * L) の各第 1 要素の符号を反転させる関数である．

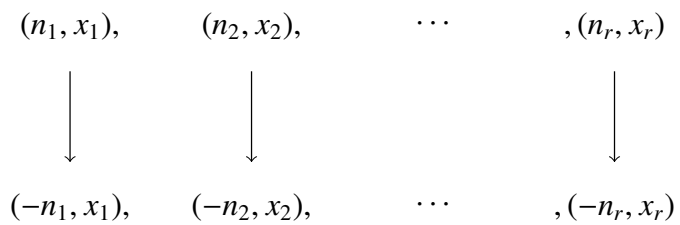

Fig. 2. minusL の出力

これを用いれば，

```
dot_inv c = \big[dot/bob]_(i ← minusL nx)  nupsz i.1 i.2
uniq [seq i.2 | i ← minusL nx]
0 \notin [seq i.1 | i ← minusL nx]
```

を条件に持ってくることができ，証明が進む．（下の二つは証明する必要がある．）

補題 1.11，補題 1.12 に関しては，数学的証明とほぼ変わらないので，定理のコードだけ示す．

```
Theorem Th3_2 c1 c2:
   ge (join (ctext c1) (ctext c2)) (ctext (dot c1 c2)).
Theorem Th3_3 c1 c2:
  join (ctext c1) (ctext (dot c1 c2))
               = join (ctext c2) (ctext (dot c1 c2)).
```

可積分は以下のように定義した．

```
Definition cintegrable x c := ge (ctext c) x.
```

これを用いて，補題 1.13 は以下のように書ける．

```
Lemma also_integ x y c cy :
  cintegrable x c ∧ meet y x = bot ∧ ge y (ctext cy)
                                → cintegrable x (dot cy c).
```

この証明も，数学的証明とほぼ同じように証明できている．

以上のことを踏まえ cain の形式化をしていく．

```
Parameter coin_margin : L → coins → coins.
Notation "c /d x" := (coin_margin x c) (at level 50).
Axiom margin0 : ∀ x c,
  cintegrable x c → ctext (c /d x) = meet (ctext c) (com x).
Axiom margin1 : ∀ x y,
  cintegrable x (up y) → (up y) /d x = up (meet y (com x)).
Axiom margin2 : ∀ x1 x2 c1 c2,
   meet x1 x2 = bot ∧
   meet x1 (ctext c2) = bot ∧
   meet x2 (ctext c1) = bot ∧
   ge (ctext c1) x1 ∧
   ge (ctext c2) x2 ∧
   cintegrable (join x1 x2) (dot c1 c2) →
   (dot c1 c2) /d (join x1 x2) = dot (c1 /d x1) (c2 /d x2).
Axiom margin3 : ∀ c, c /d bot = c.
```

$\int_x$ に当たるものを coin_margin とし，表記を合わせるため Notation にて L の元 x で coin を表す c を周辺化する際には c /d x とした．

$\int_x$ の定義が margin0 に当たる．$\int_x$ の定義には引数に取る coin が x 可積分であることが必要であるため，それを表す cintegrable x c を条件として入れている．そのため，今回は Definition ではなく Axiom として定義を形式化した．margin1 から margiin3 が $\int_x$ の 3 つ性質を表す．

以上のようにすることで，同時確率密度関数の積分を表現する代数構造 cain を形式化した．では実際に，これらを用いることで cain に関する定理が証明可能であるのか次節で見ていく．

3.3 cain の性質

cain についての主な定理を形式化していく．以下，定理 1.16 から順に示す．

定理 1.16

```
Theorem const_margin x y c cx :
  cintegrable x cx ∧
  meet x y = bot ∧
  ge y (ctext c) →
  dot c cx /d x = dot c (cx /d x).
Proof.
  rewrite /cintegrable; move⇒ []⇒ H1⇒ H2.
  rewrite -[in RHS](@margin3 c).
  rewrite -[in RHS]margin2; last first.
  apply :conj; first by rewrite meet_bot_unitL.
  apply :conj; first by rewrite meet_bot_unitL.
  apply :conj; first by[].
  apply :conj; first by rewrite /ge join_bot_unitR.
  apply :conj; first by[].
  apply :(@also_integ (join bot x) (ctext c) cx c);
  rewrite join_bot_unitL /cintegrable.
  apply :conj; first by[].
  apply :conj; first by rewrite meet_commutative.
  by rewrite /ge join_idempotent.
  by rewrite join_bot_unitL.
Qed.
```

以下は，後に必要になる分配性を持つ有界可補束の補元に関する性質の形式化になる．

```
Lemma com_join_P x y :
  complement_P (join x y) (meet (com x) (com y)).

Lemma com_meet_P x y :
 complement_P (meet x y) (join (com x) (com y)).
```

```
Lemma com_join x y : com (join x y) = meet (com x) (com y).

Lemma com_meet x y : com (meet x y) = join (com x) (com y).

Lemma double_com x : com (com x) = x.

Lemma com_bot_top_P : complement_P bot top.

Lemma com_bot_top : com bot = top.

Lemma com_top_bot : com top = bot.
```

補題 1.17

```
Lemma meet_eq_ge x y :
  meet x y = bot ↔ ge (com x) y.
Proof.
  split.
  -move⇒ H; rewrite/ ge; apply/ eqP; move: (com_meet x y);
  rewrite H com_bot_top; move⇒ H1.
  by rewrite -[in RHS](join_bot_unitL (com x)) -com_top_bot H1
             com_join !double_com [in RHS]join_commutative
             join_distributive [_ _ x]join_commutative com_top
             meet_top_unitL.
  -by rewrite/ ge; move /eqP ⇒ H1;
  rewrite -(double_com x) -(double_com y) -com_join -H1
          -join_associative com_top join_top_unitR com_top_bot.
Qed.
```

定理 1.18 とそのための補題.

```
Lemma gt_not_bot a b : gt a b → a ≠ bot.

Theorem Th6 x y z :
  gt x z ∧ meet y z = bot → mix x y /d z = mix (meet x (com z)) y.
Proof.
  case; rewrite /gt /ge; move /andP ⇒[]/eqP ⇒ H1 H2.
  -move⇒ H3; rewrite mix_up_down; last first.
  apply/(@gt_not_bot x z); rewrite/gt /ge; apply/andP; split.
  by apply/eqP. by[].  rewrite dotC const_margin; last first.
      -rewrite/cintegrable /ge ctext1 ctext4. apply: conj.
      -by rewrite [_ _ y]join_commutative -join_associative H1.
      -by rewrite meet_commutative.
       rewrite margin1; last first.
```

```
      -by rewrite /cintegrable /ge ctext1 [_ _ y]join_commutative
                  -join_associative H1.
  rewrite meet_commutative meet_distributive.
  rewrite meet_commutative in H3.
  move: H3; move/meet_eq_ge; rewrite ge_meet_def;
  move/ eqP⇒ H4; rewrite H4.
  rewrite dotC -mix_up_down ; first by rewrite meet_commutative.
  apply: meet_not_bot. exact z.
  by rewrite /ge [_ _ z]join_commutative com_top join_top_unitL.
  by rewrite /gt; apply/andP; split;
  first by rewrite /ge; apply /eqP.
Qed.
```

定理 1.19

```
Theorem Th7_1 x : up x /d x = bob.
Proof.
  rewrite margin1; first by rewrite com_bot up_bot_bob.
  by rewrite /cintegrable /ge ctext1 join_idempotent.
Qed.

Theorem Th7_2 x y : x ≠ bot → mix x y /d meet x (com y) = bob.
Proof.
  move⇒ H; rewrite mix_up_down; last by[].
  rewrite -(meet_top_unitR (join x y)) -(com_top y)
          join_commutative -join_distributive dotC const_margin;
  first rewrite margin1;
  first by rewrite com_meet double_com join_distributive
                   com_top meet_top_unitR
                   [_ _ y]join_commutative -join_distributive
                   com_bot join_commutative
                   join_bot_unitL dotC up_down_unitL.
  by rewrite /cintegrable /ge ctext1 -join_associative
             join_idempotent.
  apply: conj.
  by rewrite /cintegrable ctext1 /ge -join_associative
             join_idempotent.
  by rewrite ctext4 -meet_associative [_ _ y]meet_commutative
             com_bot meet_bot_unitR.
Qed.
```

以下は系 1.20 の (i) から (iv) である.

```
Corollary Co1_1 c y :
```

```
  meet (ctext c) y = bot → dot c (up y) /d y = c.
Proof.
  move⇒ H; rewrite const_margin; first rewrite margin1;
       first by rewrite com_bot up_bot_bob bob_unitR.
       rewrite /cintegrable ctext1; apply: ge_reflexive.
  apply: conj; first rewrite/ cintegrable ctext1;
  first apply: ge_reflexive.
  by rewrite meet_commutative.
Qed.

Corollary Co1_2 y z c :
  meet (ctext c) z = bot ∧
  meet y z = bot →
  dot c (up (join y z)) /d z = dot c (up y).
Proof.
  case⇒ H1 H2; rewrite const_margin;
  first rewrite margin1;
  first rewrite meet_commutative meet_distributive
                [_ _ z]meet_commutative com_bot join_bot_unitR;
        move: H2; rewrite meet_commutative ⇒/meet_eq_ge;
        rewrite ge_meet_def⇒/ eqP⇒ H2; first by rewrite H2.
  by rewrite /cintegrable /ge ctext1 -join_associative
             join_idempotent.
  apply: conj;
  first by rewrite /cintegrable ctext1 /ge
                   -join_associative join_idempotent.
  by rewrite meet_commutative H1.
Qed.

Corollary Co1_3 x y z c :
  meet x z = bot ∧ meet y z = bot ∧ meet (ctext c) z = bot →
  dot c (up (join x (join y z))) /d z = dot c (up (join x y)).
Proof.
  case⇒ H1 [] H2 H3; rewrite const_margin.
  rewrite margin1.
  rewrite join_associative meet_commutative meet_distributive
          [_ _ z]meet_commutative com_bot join_bot_unitR
          meet_commutative.
  have H: meet z (join x y) = bot.
  -by rewrite meet_distributive meet_commutative
              [_ _ y]meet_commutative H1 H2 join_bot.
  by move: H; move /meet_eq_ge; rewrite ge_meet_def; move /eqP;
```

```
    rewrite meet_commutative⇒ H4; rewrite H4.
  by rewrite join_associative /cintegrable /ge ctext1
             -join_associative join_idempotent.
  rewrite meet_commutative in H3; apply: conj; last by[].
  by rewrite /cintegrable ctext1 join_associative /ge
             -join_associative join_idempotent.
Qed.

Corollary Co1_4 x y z :
  x ≠ bot ∧ meet x y = bot ∧ meet y z = bot →
  mix (join x y) z /d y = mix x z.
Proof.
  case ⇒ H1 [] H₂ H3; rewrite mix_up_down;
  first rewrite dotC const_margin.
  rewrite margin1.
  rewrite -join_associative [_ _ z]join_commutative
          join_associative meet_commutative meet_distributive
          [_ _ y]meet_commutative com_bot join_bot_unitR
          meet_commutative.
  have H: meet y (join x z) = bot.
   -by rewrite meet_distributive meet_commutative H₂ H3 join_bot.
  by move: H; move /meet_eq_ge; rewrite ge_meet_def; move /eqP;
     rewrite meet_commutative⇒ H4; rewrite H4 dotC -mix_up_down.
  by rewrite /cintegrable /ge ctext1 -join_associative
             [_ z y]join_commutative join_associative
             -[_ (join x y) y]join_associative join_idempotent.
  apply: conj.
   -by rewrite /cintegrable ctext1 -join_associative
               [_ y _]join_commutative join_associative /ge
               -join_associative join_idempotent.
   -by rewrite ctext4.
  have H: (x ≠ bot) || (y ≠ bot) → join x y ≠ bot.
   -rewrite -negb_and; apply: contra; apply/join_eq_bot.
  by apply /H /orP /or_introl.
Qed.
```

定理 1.21

```
Lemma ctext_eq_ge x c :
  meet (ctext c) (com x) = bot ↔ ge x (ctext c).

Lemma N_law2_1 x y z cx cy :
  meet (ctext cx) x = bot ∧
```

```
  meet (ctext cy) y = bot ∧
  up (join (join x y) z) = dot cy cx →
  up (meet (join y z) (com x)) = dot cx (cy /d x).

Lemma dot_eq c1 c2 c3 : c1 = c2 ↔ dot c1 c3 = dot c2 c3.

Lemma N_law2_2 x y z cx cy :
  meet x y = bot ∧
  meet (ctext cx) x = bot ∧
  meet (ctext cy) y = bot ∧
  up (join (join x y) z) = dot cy cx →
  up (meet z (meet (com x) (com y))) = dot (cx /d y) (cy /d x).

Theorem N_law2 x y z cx cy :
  meet x y = bot ∧
  meet (ctext cx) x = bot ∧
  meet (ctext cy) y = bot ∧
  up (join (join x y) z) = dot cy cx →
  up (join (join x y) z) =
  dot (dot (up (meet (join x z) (com y)))
           (up (meet (join y z) (com x))))
      (down (meet (meet z (com x)) (com y))).
Proof.
  case⇒ H1 [] H2 [] H3 H4.
  rewrite (@N_law2_1 y x z cy cx);
  last by rewrite [_ y _]join_commutative dotC.
  rewrite (@N_law2_1 x y z cx cy); last by[].
  rewrite [_ cy _]dotC dotA -[_ _ cx]dotA [_ (cx /d y) _]dotC
          -2!dotA -H4 [_ (cx /d y) _]dotA -meet_associative
          -(@N_law2_2 x y z cx cy); last by[].
  by rewrite (@up_down_unitL (meet z (meet (com x) (com y)))) dotr0.
Qed.

Theorem N_law2' x y z cx cy :
  meet y z = bot → meet x z = bot →
  meet x y = bot ∧
  meet (ctext cx) x = bot ∧
  meet (ctext cy) y = bot ∧
  up (join (join x y) z) = dot cy cx →
  up (join (join x y) z) =
  dot (dot (up (join x z)) (up (join y z))) (down z).
Proof.
```

```
  move⇒b c h. move:(b)(c)(h)⇒B C H.
  apply N_law2 in H.
  move:h⇒[]⇒a⇒h;move:(a)⇒A.
  rewrite meet_commutative in A.
  apply meet_eq_ge in A;apply meet_eq_ge in B;apply meet_eq_ge in C.
  rewrite ge_meet_def in A;
  rewrite ge_meet_def in B;
  rewrite ge_meet_def in C.
  apply meet_eq_ge in a;rewrite ge_meet_def in a.
  move /eqP in A; move/eqP in B; move/eqP in C; move/eqP in a.
  rewrite meet_commutative meet_distributive A B in H.
  rewrite meet_commutative meet_distributive a C in H.
  by rewrite (meet_commutative z _) C meet_commutative B in H.
Qed.
```

上記の N_law2 が，定理 1.21(i) の形式化であり，N_law' が定理 1.21(ii) の形式化である．この 2 つの定理に 3 つの補題を作成した．

以上が cain に関する定理の形式化である．

3.4 因数分解定理の形式化

```
Definition co_ind x y z := (mix x (join y z)) = (mix x z).
```

条件付き独立の定義である $\mathbb{T}^x_{y\vee z} = \mathbb{T}^x_z$ なる等式に co_ind と名前をつけることで形式化した．

定理 1.23

```
Lemma join_bot_bot x y:
  x ≠ bot → y ≠ bot → (join x y) ≠ bot.

Theorem co_ind_eq_2 x y z:
  x ≠ bot →
  y ≠ bot →
  (co_ind x y z ↔
  (up (join (join x y) z) = dot (up (join y z)) (mix x z))).
Proof.
  move⇒XH YH. rewrite/co_ind.
  rewrite mix_up_down⇒//. rewrite div_up_down.
  by rewrite -join_associative dotrC.
Qed.
```

以上の形式化と次の補題により，因数分解定理を表す定理 1.24 の形式化 co_fact が作られる．

```
Lemma ge_mix_bob_eq x y:
  x ≠ bot → (ge y x ↔ (mix x y = bob)).

Theorem co_fact x y z:
  x ≠ bot →
  y ≠ bot →
  z ≠ bot →
  meet x y = bot →
  meet y z = bot →
  meet x z = bot →
  (co_ind x y z ↔
  ∃ cx cy,
    ge (com x) (ctext cx) ∧
    ge (com y) (ctext cy) ∧
    up (join x (join y z)) = dot cx cy).
Proof.
  move⇒X Y Z A B C.
  split⇒ H.
  apply co_ind_eq_2 in H⇒//.
  ∃ (up (join y z)), (mix x z).
  split.
  apply/meet_eq_ge.
  by rewrite ctext1 meet_distributive A C join_idempotent.
  split.
  apply/meet_eq_ge.
  move:(excluded_middle_axiom (ge z x));case⇒Hzx.
  apply ge_mix_bob_eq in Hzx⇒//.
  by rewrite Hzx ctext1 meet_bot_unitR.
  move/negP in Hzx.
  rewrite ctext3⇒//.
  by rewrite meet_distributive B meet_commutative A join_idempotent.
  by rewrite join_associative.

  apply /co_ind_eq_2⇒//.  move :H⇒[] cx [] cy.
  case⇒ E []⇒ F⇒ G.
  rewrite mix_up_down//.  rewrite dotrA (dotrC (up (join y z)) _).
  apply /(@N_law2' x y z cx cy)⇒//.  split⇒//.
  split; first by rewrite meet_commutative meet_eq_ge.
  split; first by rewrite meet_commutative meet_eq_ge.
  by rewrite -join_associative dotrC.
Qed.
```

以上により，統計学における因数分解定理の形式化が完了した．

3.5 graphoid 性

次に cain による条件付き独立の定義が graphoid の公理を満たしていることの根拠となる定理 1.31~1.35 について以下のようにした．

```
Theorem co_sym x y z :
 x ≠ bot → y ≠ bot → (co_ind x y z ↔ co_ind y x z).

Theorem co_decomp x y z a b:
 a ≠ bot → b! = bot →
  meet x y = bot → co_ind x y z →
  ge x a → ge y b → co_ind a b z.

Theorem co_w_u x y z w:
  x ≠ bot → y ≠ bot → z ≠ bot → w ≠ bot→
  meet x y = bot → meet x z = bot →
  meet x w = bot → meet y z = bot →
  meet y w = bot → meet w z = bot →
  co_ind x (join y z) w → co_ind x y (join z w).

Theorem co_contr x y z w:
  x ≠ bot → y ≠ bot → z ≠ bot → w ≠ bot→
  meet x y = bot → meet x z = bot →
  meet x w = bot → meet y z = bot →
  meet y w = bot → meet w z = bot →
  (co_ind x y (join z w) ∧ co_ind x z w
 ↔ co_ind x (join y z) w).

Theorem co_inter x y z w:
  x ≠ bot → y ≠ bot → z ≠ bot → w ≠ bot →
  meet x (com z) ≠ bot → meet x y = bot → meet x z = bot →
  meet x w = bot → meet y z = bot →
  meet y w = bot → meet w z = bot →
  (co_ind x (join y z) w ↔
  (co_ind x y (join z w)∧co_ind x z (join y w))).
```

これら 5 つの定理はどれも基本的には紙の上での証明になぞらえて `M_law` や `N_law` を適用させてやればよいが，最後の `co_inter` の証明の時のみ，大小関係の場合分けのために排中律を使っている．また，この 5 つの定理が形式化されたことから cain によって条件付き独立性を表現したものが graphod 性を備えていることが確実に言えた．次に

(strong)separoid 性については以下のようにまとめて形式化した．

```
Theorem co_ind_sep x y z w:
  x ≠ bot → y ≠ bot → z ≠ bot → w ≠ bot →
  meet x y = bot → meet x z = bot →
  meet x w = bot → meet y z = bot →
  meet y w = bot → meet w z = bot →
    (co_ind x y x )
  ∧ (co_ind x y z → co_ind y x z)
  ∧ (co_ind x y z → ge y w → co_ind x w z)
  ∧ (co_ind x y z → ge y w → co_ind x y (join z w))
  ∧ (co_ind x y z → co_ind x w (join y z) → co_ind x (join y w) z)
  ∧ (ge y z → ge y w → co_ind x y z →
    co_ind x y w → co_ind x y (meet z w))).
```

証明に関しては定理 1.31~1.35 と以下に記した Theorem co_id を使えば，あとは coin の簡単な変形を繰り返すだけで完了させることができる．これで (strong)separoid 性についても形式化することができた．

```
Theorem co_id x y:x ≠ bot → co_ind x y x.
```

4. 今後の課題

4.1 context の形式化

context の形式化は，まだ考察の余地がある．例えば，今の定義は

```
Parameter ctext : coins → L.

Axiom weak_ctext: ∀ (c:coins) (nx: seq(int*L)),
   c = \big[dot/bob]_(i ← nx)  nupsz i.1 i.2 →
   uniq [seq i.2 | i ← nx] →
   0 \notin [seq i.1 | i ← nx] →
   ctext c = \big[join/bot]_(i ← nx) i.2.
```

のようになっているが，もともと ctect は coin_prod_up_strong の性質を用いて定義されているものなので，形式化の定義にも coin_prod_up_strong を入れて定義するべきであり，したがって，以下のように定義するのが本当は自然である．

```
Definition coin_xns (c:coins)
      := sval (coin_prod_up_strong c).
Definition ctext (c:coins):=
      \big[join/bot]_(i ← (coin_xns c)) i.2.
```

この定義は，dot_inv を定義したときと同様の方法を用いている．この定義ならば，元 c に対する分解で，生成された lattice の元の結びを，直接扱うことができる．しかし，こちらの定義による証明は現段階では完成していないので，これからの課題と言えるだろう．

4.2 prime

本論文は，Wang 氏の論文 [1] を参考に形式化を行ったが，形式化するにあたって，本論文と Wang 氏の論文とにギャップが生まれてしまっているのでその点についてこの節では述べる．

本論文で何度も使った定理 1.7 の canonical expression という表記だが本来はまず，その前に以下のような prime 性について定されている．

定義 4.1 (prime) rasing coin $\mathbb{T}^x$ が prime であるとは，

$$\mathbb{T}^x = \left(\mathbb{T}^{x_1}\right)^{n_1} \cdots \left(\mathbb{T}^{x_r}\right)^{n_r} \quad (x_i < x)$$

のような分解を持たないこと．

さらに，Lattice $\mathbb{L}$ に DCC（descending chain condition）を仮定する．

定義 4.2 (降鎖条件 (descending chain condition)) 半順序集合 X において，任意の下降列 $x_1 > x_2 > \cdots$ が有限回で止まるとき，X は降鎖条件を満たすという．

そして，これらを用いて，

定理 4.3 $\mathbb{L}$ は DCC を満たす束とする．$\forall \mathbb{T} \in \mathfrak{C}$ に対し，

$$\mathbb{T} = \left(\mathbb{T}^{x_1}\right)^{n_1} \cdots \left(\mathbb{T}^{x_r}\right)^{n_r}$$

という分解が存在する．但し，各 x_i は互いに異なり，かつ，各 $\mathbb{T}^{x_i}$ は prime で，各 $n_i \neq 0$ である．このように分解された形を canonical expression という．

のように canonical expression を定義している．この定理で課題なのは，どのようにして prime 性を判定するのか，ということである．DCC 自体は Coq/SSReflect 内の well_founded を用いて簡単に定義できるのだが，どうしても分解が止まる止まらないの判断の時点で，prime 性の判定をしなければならない．prime 性を用いた定義だと，coin の分解が一意に定まり，仮定などせずとも当然 context も一意に定まる．案としては，Program Fixpoint を用いる方法や，まだデフォルトで mathcomp に組み込まれていないライブラリ finmap（多重集合に関するライブラリ）を使う方法が考えられるが，残念ながらこれに関しては，手を付けられていないのが現状である．

4.3 order.v

前説で述べた finmap は将来 mathcomp に実装される期待のかかったライブラリであるが，その中に order.v というファイルがある．これは，順序に関してのライブラリで，本論文の lattice の形式化や Lemma に関して有効に利用できると考えている．

参考文献

[1] Wang Jinfang, “A Universal Algebraic Approach for Conditional Independence, ” Ann. Inst. Stat. Math. 62(4), 747-773 (2010)

[2] The Coq Proof Assistant, https://coq.inria.fr, INRIA

[3] Mathematical Components, http://math-comp.github.io/math-comp/

[4] Programs and Proofs, http://ilyasergey.net/pnp/pnp.pdf

[5] Reynald Affeldt, “定理証明支援系 Coq による形式検証,”
https://staff.aist.go.jp/reynald.affeldt/ssrcoq/
集中講義. 京都大学大学院理学研究科数学・数理解析専攻数理解析系, Jul. 2015.

[6] Gontheir, G., Mahboubi, A. , “An Introduction to small scale reflection in Coq” J. Autom. Reasoning., 3, 95-152 (2010).

[7] Gontheir, G., Mahboubi, A.,Tassi, E. , “A small scale reflection extension for the Coq system.” Version 10. Thechnical Report RR-6645. INRIA (2011)

[8] Manabu Hagiwara, Wang Jinfang, Mitsuharu Yamamoto, “Formalization of probabilistic conditional independence relation using Coq/SSreflect ”