プログラム意味論とトポロジー
再帰・相互作用・結び目

長谷川　真人　(京都大学数理解析研究所)*

1. はじめに：計算機科学と数学の関係について

1980年代末に，数学者Peter Freydは，計算機科学と数学の関係について以下のように述べている [7]：

> . . . *Computer Science Contradicts Mathematics.* Many modern programming languages are inconsistent with standard mathematical foundations. The task of finding sound interpretations for what it is that computer scientists do strikes this writer as, perhaps, the highest type of applied mathematics. It is akin to the process that has been going on throughout the 20th Century with respect to physics. The interaction between the mathematicians and the practitioners in each case has resulted in the growth of both subjects. . . .

これは，1970年頃にDana Scottらにより創始された，**領域理論**（domain theory）を基盤とするプログラミング言語の**表示的意味論**（denotational semantics）[2, 29, 35] を念頭に置いたものである。プログラミング言語においては，自己言及，相互参照，自己適用，大域的なジャンプ，局所状態とブロック構造，多相性，等々，素直に数学的に定式化することが困難な事柄が山のようにある．領域理論と表示的意味論は，それらの諸問題の多くについて成功を収めた．Freydいわく，その領域理論で用いられる数学は，通常の数学の常識からはいささか異なるものである．Freydは，[7] を含む一連の論文で，その計算機科学固有の事情を，簡潔な圏論の言葉で特徴付けた．Freydの仕事を発展させたものが公理的領域理論（axiomatic domain theory）として1990年代に英国を中心に盛んに研究され，これはある意味で領域理論という分野の総仕上げとも言えるものであった．

Freydの言っていることは，我々計算機科学者にとって，大いに励みに，また刺激になるものである．筆者も，計算機科学において必要とされる数学は，必ずしもメインストリームの数学において筋が良いとされるものではない，とは常々感じている．計算機科学の研究対象の多くは，離散的で，非可換で，非線形で，非対称である．事態をややこしくする要因でてんこ盛りである．しかし，だからといって普通の数学のノウハウが計算機科学において全然使えないというわけでは決してない．たとえば，領域理論においては，計算に内在するある種の連続性に着目し，データ構造をある種の位相空間として，またデータを処理するプログラムを位相空間の間の連続写像として捉える．これは，複雑な離散的な現象を，

本研究は科研費 (課題番号:20500010) の助成を受けたものである。
2010 Mathematics Subject Classification: 68Q55, 18D10, 68N18, 03B70, 57M25
キーワード：プログラム意味論, モノイダル圏, 相互作用の幾何，結び目理論，量子不変量
* 〒606-8502 京都市左京区北白川追分町　京都大学 数理解析研究所
e-mail: hassei@kurims.kyoto-u.ac.jp
web: http://www.kurims.kyoto-u.ac.jp/~hassei/

連続的な簡明な構造を用いて上手に扱ってみせた，格好の例といって良い．別の例では，Girard の**線形論理**（linear logic）[9] （より正確にはその表示的意味論）は，非可換・非線形・非対称な計算の世界を，可換・線形・対称な計算プラスアルファとして（もちろんそのプラスアルファのところが肝心なのであるが）筋よく分解するものである [32].

だから，Freyd の主張にあえて異議を申し立てるなら，計算機科学の数学というのは，主流派の筋の良い数学と対立・矛盾するものではなく，どちらかといえば主流派の数学にプラスアルファ何かを追加するかたちで展開されるものというのが，より実情に沿った見方ではないかと思う．いや，むしろ，こういう見方が普通にできるようになってきたことが，1980 年代以降の計算機科学に起きたある種の成熟の結果というべきかも知れない．主流派の数学の成果を素直に享受できるほどに，計算機科学が大人になってきたということではないだろうか．

この講演で触れる話題も，そのような，数学から計算機科学へのテクノロジー・トランスファーの一例である．数理物理，表現論，低次元トポロジーといった分野で培われた数学が，プログラム意味論の分野にどのような影響を与えてきたかを紹介し，その意義と将来の展望について考察する．

2. プログラム意味論

プログラムは，それ自体は有限長の記号の列に過ぎない．プログラムの表現する計算＝プログラムの**意味**（semantics）については，別に定義する必要がある．この，プログラムの意味について研究するのが**プログラミング言語の意味論**（**プログラム意味論**）である．現在では，プログラミング言語の意味論には大きく分けてふたつのアプローチがある．実際のプログラミング言語の処理系が行っているように，プログラムの構文の構造に即してプログラムの意味を（多くの場合具体的に実行可能な計算方法として）与えるのが**操作的意味論**（operational semantics），それに対し，プログラムの表わす計算を適切な数学的対象として抽象化して表現し調べるのが表示的意味論である．

プログラム意味論の問題設定はおおむね以下のようなものである．まず，プログラミング言語を一つ固定し，そのプログラム全体の集合を $\mathbf{P}$ で表わすことにする．$\mathbf{P}$ の要素を適当な文法により具体的に記述するのはプログラミング言語の構文論（syntax）の話題であるが，ここではその詳細には立ち入らない．次に，プログラムの実行の結果の全体を $\mathbf{V}$ で表わす.「プログラムの実行結果」というのも本当はいろいろな可能性があり，その定式化は自明ではないが，ここでは簡単のため $\mathbf{V}$ は $\mathbf{P}$ の部分集合であるものとする．すなわち,「実行結果」を「これ以上計算を行う必要のないプログラム」と同一視する．そして，プログラムの実行は，$\mathbf{P}$ 上の二項関係 $\to \subseteq \mathbf{P} \times \mathbf{P}$ により定める．プログラム P，P' について，$P \to P'$ であるとは，P が定める計算を一ステップ実行したあとの（多くの場合中間的な）結果が P' であることを表わす．$\to$ の具体的な定義は適切な操作的意味論により規定される．現実のプログラミング言語であれば，その処理系により規定されると思ってもよい．そして，プログラム $P \in \mathbf{P}$ の実行結果が $V \in \mathbf{V}$ であることを，

中間結果の有限列 $P_1, P_2, \ldots, P_n$ が存在して

$$P \to P_1 \to P_2 \to \ldots \to V$$

となることと定める．プログラミング言語の理論やラムダ計算に馴染みのある方であれば，具体的な例として，$\mathbf{P}$ を（自由変数を含まない）ラムダ項全体の集合，$\to$ をラムダ項上の適当な簡約関係（β 簡約や $\beta\eta$ 簡約，あるいは正規順序などの評価戦略），$\mathbf{V}$ を $\to$ に関する正規形と思っていただいて構わない．

以上の状況について，様々な基本的な問題が考えられる．まず，プログラムの実行が正常に停止して実行結果が得られるのはどのような場合か，という問題が自然に考えられる．また，プログラムの実行が一意に定まるかどうか，さらにプログラムが途中で（計算結果にたどり着くことなく）異常終了したりしないか，なども基本的な問題である．さらには，実行結果が満たす性質（あるいは満たすべき仕様）を検証するという問題も重要である．

それらの諸問題のなかでも，プログラムの**等価性**，すなわち与えられたふたつのプログラム $P, P' \in \mathbf{P}$ が等しいかどうか，という問題は最も基本的なものである．ただ，この「等しさ」にも実はいろいろな選択肢があり，どのような応用を念頭に置いているかで適宜使い分けられるのだが，ここでは，上述の実行関係 $\to$ を含む最小の同値関係 $=$ に関してプログラムの等しさを議論することにする．これは実用上重要なプログラム上の同値関係の中では，最強（＝最大）のものではないものの，最も由緒正しく，また汎用性の高い同値関係である．上述のラムダ計算の例では，$\to$ が β 簡約関係なら $=$ は β 同値関係である，

しかし，一般に，$P = P'$ であるかどうかを正確に捉えるのは困難である．まともな（Turing 完全な）プログラミング言語については，これらの等価性は決定不可能である（プログラムの等価性を決定するプログラムは書けない）．そこで考えられるのが表示的意味論である．表示的意味論では，$\mathbf{P}$ から何らかの集合 $\mathbf{D}$ への写像（**意味関数**と呼ばれ，通常 $[\![-]\!]$ で表わす）であって，$P = P'$ ならば $[\![P]\!] = [\![P']\!]$ となるものを，このプログラミング言語のモデルとして考える．特に $P \to P'$ なら $[\![P]\!] = [\![P']\!]$ であるので，モデルを与えることは計算の実行に関する不変量を与えることであるともいえる．

もちろん，$\mathbf{P}$ を $=$ で割った商集合を $\mathbf{D}$ とし，$[\![P]\!]$ を P の同値類とすれば，自明な表示的意味論が得られる（具体的なプログラム（項）の商集合として得られるので，しばしば**項モデル**と呼ばれる）．しかし，この自明な表示的意味論から新たに得られる情報はほとんどない．本当に求められているのは，プログラミング言語の構文等に依存することなく，独立した数学的な（可能ならば扱いやすい）対象として得られる $\mathbf{D}$，およびある種の構造を保存する写像として得られる意味関数 $[\![-]\!]$ である．そのようなモデルを使って，プログラムの等価性について有用な情報を得ることができる．特に，$[\![P]\!] \neq [\![P']\!]$ であることがわかれば，ただちに $P \neq P'$ であることがわかる．$[\![P]\!] = [\![P']\!]$ である場合には $P = P'$ かどうかは一般にはわからないが，モデルの持つ性質を考慮に入れることにより $P = P'$ であると結論できる場合もある．

理想的には，このようなプログラミング言語のモデルは，なんらかの代数系として定式化され，項モデルは自由生成された代数，そして$[\![-]\!] : \mathbf{P} \to \mathbf{D}$は項モデルから$\mathbf{D}$への準同型写像として与えられるというのが望ましい.（ラムダ計算については，実際にそのような定式化が整備されている.）また，ここまで簡単のため$\mathbf{P}$（$\mathbf{P}/=$）や$\mathbf{D}$は単なる集合としてきたが，実際的なプログラミング言語，特にデータ型の概念のあるプログラミング言語については，これらを単なる集合ではなく何らかの構造の入った圏と見なし，$[\![-]\!]$は自由生成された圏からの構造を保存する関手により定まる，とするのがより自然である．実際，型付きラムダ計算（$\to$は$\beta\eta$簡約関係）については，その表示的意味論は有限直積および冪対象を持つ圏（**カルテジアン閉圏**，cartesian closed category）と，それらの構造を保つ関手により明快に展開される [25, 4]．型付きラムダ計算の項モデルは自由生成されたカルテジアン閉圏と同値であるので，型付きラムダ計算のモデルを定めることは，カルテジアン閉圏（と生成元の像）をひとつ与えることに他ならない.

現在では，プログラム意味論の多くが，このようなfunctorialな意味論のかたちで扱えるよう整備されてきている．そこで用いられている圏論的構造には，上述のカルテジアン閉圏のみならず，ぱっと思いつくものだけでもトポス，層，ファイブレーション，2-圏・双圏（bicategory），豊穣圏（enriched category），そして（様々な構造が付加された）モノイダル圏（テンソル圏）などが挙げられる．例えば，領域理論には，ここに列挙した構造のほとんどすべてが登場する．公理的領域理論では，カルテジアン閉圏およびそれから導かれるモノイダル圏で（通常の数学の感覚では理不尽なくらいに強い）ある種の閉包性を満たすものがモデルである（その閉包性は豊穣圏の概念を用いて定義される）．以下では，特に，ある種の，量子トポロジーに由来するモノイダル圏（リボン圏とトレース付きモノイダル圏）に注目し，そのプログラム意味論における役割について見ていく.

3. 量子トポロジーとモノイダル圏

Jones多項式の発見を契機に，1980年代から，結び目や3次元多様体の不変量が，数理物理，統計物理，量子群とその表現論など，幅広い分野の量子論的な手法を用いて大量に導入された．これらは**量子不変量** [33, 28] と呼ばれ，その研究分野を**量子トポロジー**と呼ぶ.

いうまでもなく，トポロジーにおける不変量とは，幾何学的な対象（たとえば結び目）から適当な集合への写像であって，その幾何学的対象の連続的な変形（イソトピー）に対して不変なもののことである．その使いみちは，基本的にはこれまでに述べたプログラミング言語の表示的意味論のそれと同じである．表示的意味論がプログラムの実行に関する不変量を考えるのに対し，結び目の理論では，結び目の連続変形に関する不変量を考えるというわけで，発想は全く同じといって良い.

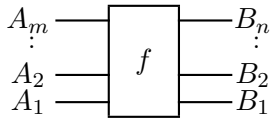
もっとも，不変量の考え方自体は数理科学のほぼ全分野に共通するものであるから，アイデアや使いみちが一緒だと言っても驚くほどのことではない．それでは，具体的に用いられている構造についてはどうであろうか．共通点を見出すた

めには，どのように比較したら良いだろうか．幸いなことに，結び目の量子不変量の理論（のかなりの部分）も，プログラム意味論と同様に，functorialな理論として圏論の枠組みを用いて理解することができる [8, 20, 33, 34]．この，低次元トポロジーへの圏論の応用は，1980年代から90年代にかけて，量子不変量の研究と呼応するかたちで発展した[1]．ここで中心的な役割を果たすのは，リボン圏をはじめとする，適当な構造の入ったモノイダル圏である．

モノイダル圏（monoidal category）あるいは**テンソル圏**（tensor category）[26, 18] $\mathbf{C} = (\mathbf{C}, \otimes, I, a, l, r)$ とは，圏 $\mathbf{C}$ と，それに付随する**テンソル積**または**モノイダル積**と呼ばれる関手 $\otimes : \mathbf{C}^2 \to \mathbf{C}$，単位対象と呼ばれる $\mathbf{C}$ の対象 I，テンソル積と単位対象の結合律・単位律に相当し適当な公理をみたす可逆な自然変換 $a_{A,B,C} : (A \otimes B) \otimes C \xrightarrow{\sim} A \otimes (B \otimes C)$, $l_A : I \otimes A \xrightarrow{\sim} A$ および $r_A : A \otimes I \xrightarrow{\sim} A$ の組のことをいう．a, l, r がすべて恒等射であるようなモノイダル圏は，厳格（strict）であるという．厳格なモノイダル圏では，$A \otimes (B \otimes C)$ と $(A \otimes B) \otimes C$ は同一の対象であり，$A \otimes I$，$I \otimes A$ と A についても同様である．任意のモノイダル圏は，厳格なモノイダル圏と（モノイダル圏の構造を厳格に保存する関手を介して）同値である（コヒーレンス定理）．したがって，モノイダル圏について考察する際には，a, l, r のことは忘れて，あたかも厳格なモノイダル圏を扱っているようにみなしても問題は起きない．以下でも，a, l, r のことは無視して話を進める．

モノイダル圏で，条件 $c_{A,B\otimes C} = (1_B \otimes c_{A,C}) \circ (c_{A,B} \otimes 1_C)$ および $c^{-1}_{B\otimes C,A} = (1_B \otimes c^{-1}_{C,A}) \circ (c^{-1}_{B,A} \otimes 1_C)$ を満たす可逆な自然変換 $c_{A,B} : A \otimes B \xrightarrow{\sim} B \otimes A$ を持つものを，**ブレイド付きモノイダル圏**（braided monoidal category）と呼び，c は**ブレイド**と呼ばれる．さらに，**バランス**あるいは**ツイスト**と呼ばれる可逆な自然変換 $\theta_A : A \xrightarrow{\sim} A$ が存在して $\theta_I = 1_I$ および $\theta_{A\otimes B} = c_{B,A} \circ (\theta_B \otimes \theta_A) \circ c_{A,B}$ をみたすブレイド付きモノイダル圏を，**バランス付きモノイダル圏**（balanced monoidal category）という．バランス付きモノイダル圏で，$\theta = 1$（したがって $c = c^{-1}$）が成り立つものは，**対称モノイダル圏**（symmetric monoidal category）と呼ばれる．

モノイダル圏の射は，簡潔に図示することができ（しばしばPenrose diagramやstring diagramと呼ばれる），それは図の連続的な変形にたいして整合的である（対応する射は不変である）[17][2]．この講演では，射 $f : A_1 \otimes A_2 \otimes \ldots \otimes A_m \to B_1 \otimes B_2 \otimes \ldots \otimes B_n$ を

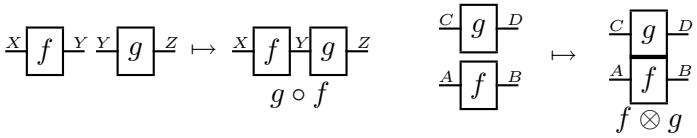


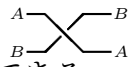
とあらわす．射の合成やテンソル積は，それぞれの射が対応する図の直列つなぎ

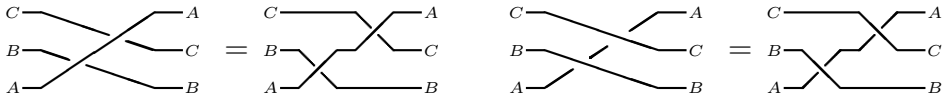
[1] Freydはこの方面でも貢献している [8]．また，彼はHOMFLY不変量のFでもある．

[2] これは，モノイダル圏を用いてプログラム意味論を展開することの大きな利点である．プログラムの意味をモノイダル圏の射として捉えることにより，ふたつのプログラムが，図示して連続変形を介して同じものであるかどうか，きちんと議論することができる．プログラムの構造を視覚的に捉える試みはこれまでにも数多くなされてきたが，このようにプログラムを結び目と同様の位相幾何的な対象として厳密に扱えるようになったことは大きな進歩であると思う．Melliès[27] は，線形論理の証明網（proof net）について同様の観点から論じている．

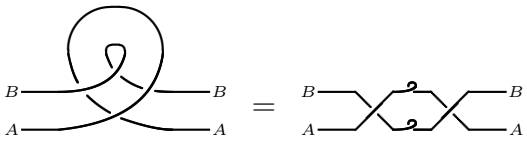
および並列つなぎであらわされる（下図）.



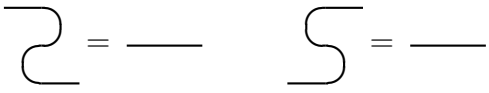
ブレイド $c_{A,B} : A \otimes B \to B \otimes A$ は交差  として，また $c_{A,B}^{-1} : B \otimes A \to A \otimes B$ は逆向きの交差  としてあらわされる．ブレイドの条件式は，以下のように自然に図示できる．



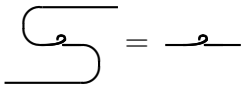
バランス $\theta_A : A \to A$ は，ねじれ  であらわされる．θ_A^{-1} は逆向きのねじれ である．バランスの条件式も以下のようにあらわせる．



最後に，モノイダル圏における双対性の概念を導入する．モノイダル圏の対象 A の**（左）双対**とは，対象 A^* および射 $\eta_A : I \to A \otimes A^*$ と $\varepsilon_A : A^* \otimes A \to I$ （それぞれ  と  で図示される）で，条件

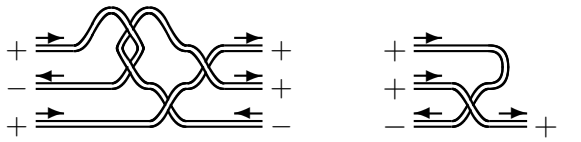


を満たすものである．**リボン圏**（ribbon category）[33, 34] あるいは**トーティルモノイダル圏**（tortile monoidal category）[30] とは，すべての対象が双対を持ち，



が成り立つようなバランス付きモノイダル圏である．なお，対称な（$\theta = 1, c = c^{-1}$ であるような）リボン圏は，しばしば**コンパクト閉圏**（compact closed category）[23] とも呼ばれる．たとえば，体 K について，K 上の有限次元線形空間を対象に，また線形写像を射に持つ圏 $\mathbf{Vect}_K^{\mathbf{fin}}$ は，対称なリボン圏（すなわちコンパクト閉圏）である．この場合，テンソル積 $U \otimes V$ は線形空間のテンソル積 $U \otimes_K V$，単位対象は K，ブレイド $c_{U,V}$ は $U \otimes V$ から $V \otimes U$ への自明な同型写像，バランスは恒等写像であり，U の双対 U^* は双対空間 $\hom(U, K)$ により与えられる．

結び目の理論からは，以下のような（向きづけられた，枠付きの）タングルの圏 **Tangle** が自然に構成できる [8, 33, 34]．**Tangle** の対象は，（タングルの向きを表わす）記号 $+$ と $-$ の有限列である．そして，$(s_1, \ldots, s_m)$ から $(s'_1, \ldots, s'_n)$ への射は，下図のような，$+$ では左から右に，$-$ では右から左に向き付けられた，$s_1, \ldots, s_m, s'_1, \ldots, s'_n$ を端点とするタングルの，連続変形に関する同値類である．



Tangle は，リボン圏である．それも，ただのリボン圏ではなく，ひとつの対象から自由生成されたリボン圏である [30]．したがって，任意のリボン圏 **C** とその対象をひとつ選ぶことにより，**Tangle** から **C** へのリボン圏の構造を保存する関手が定まり，この関手からタングルの不変量がひとつ得られる．もちろん，**C** としては非自明な（対称でない）リボン圏を選ばないと，意味のある不変量は得られない．例えば，有限次元線形空間と線形写像のなす対称リボン圏は，そのままでは役に立たない．しかし，**量子群** [5, 16]（**リボン Hopf 代数** [33]）からは，その表現の圏としてリボン圏を構成することができる．すなわち，量子群から，タングルの不変量を構成することができる．

リボン圏を経ずに，量子群から直接不変量を導出することも可能である [28]．しかし，圏論を用いた，functorial な不変量の枠組みは，プログラム意味論における functorial な表示的意味論と比較するのに大変便利である．特に，プログラム意味論において，型付きラムダ計算の意味論のためにカルテジアン閉圏が果たす役割と，結び目の理論において，タングルの不変量のためにリボン圏が果たす役割は，完全に対応していることがわかる．この観察から，もしもカルテジアン閉圏とリボン圏の両方の構造を持つ圏が存在するならば，いささか飛躍した夢想であるが，プログラムと結び目とがごちゃまぜになったような愉快な（あるいは悪夢のような）状況について議論することだってできてしまいそうである．しかし，カルテジアン閉圏とリボン圏の両方の構造を持つ圏は自明な圏だけなので，残念ながら，そんなことは実際には起こりえないように思われる．

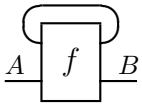
ところが，実は，リボン圏の断片，より正確にはリボン圏の充満部分モノイダル圏（テンソル積で閉じた充満な部分圏）が，プログラム意味論において，それとは知られることなく用いられていたのである．特に，領域理論においては，約 40 年前からカルテジアン閉圏であってしかもあるリボン圏の充満部分モノイダル圏であるようなものが，極めて重要な役割を果たしてきてきた．領域理論で必要とされる数学は，確かに Freyd の言うように普通の数学の感覚から外れているところもあるが，同時に，最近の数学の重要な部分をも，ある程度含んでいたのである．それが明らかになったのは，15 年ほど前，トレース付きモノイダル圏 [19] の概念が導入されてからのことである．

4. トレース付きモノイダル圏

トレース付きモノイダル圏には，二つの特徴付けがある．一つは，抽象的なトレース演算子を備えたバランス付きモノイダル圏というもので，これが Joyal, Street, Verity の原論文 [19] における定義である．もう一つは，同じ論文で与えられている構造定理によるもので，結論だけ述べると，トレース付きモノイダル圏は，必ずあるリボン圏の充満部分モノイダル圏になっている，というものである．トレー

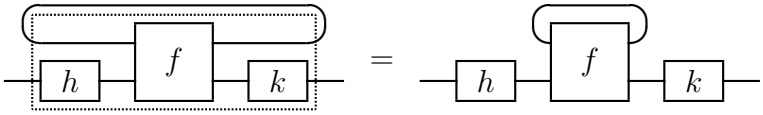
ス付きモノイダル圏では，対象の双対は必ずしも存在しない．これは，プログラム意味論にとっては朗報である．プログラム意味論で用いられる圏は，多くの場合双対性を持たないからである．構造定理については後で触れることにして，まず，前者のトレース演算子による特徴付けから説明しよう．

いま，バランス付きモノイダル圏 $\mathbf{C}$ が与えられているものとする．このとき，$\mathbf{C}$ 上の**トレース**（trace）演算子 [19] とは，$\mathbf{C}$ の対象で添字付けられた関数の族 $Tr^X_{A,B} : \mathbf{C}(A \otimes X, B \otimes X) \to \mathbf{C}(A, B)$ で，以下の四つの条件を満たすものである．ここで，$f : A \otimes X \to B \otimes X$ のトレース $Tr^X_{A,B}(f) : A \to B$ を



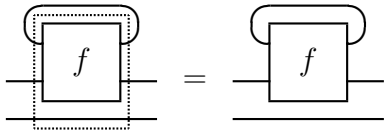
であらわす．

$$Tr^X_{A',B'}((k \otimes 1_X) \circ f \circ (h \otimes 1_X)) = k \circ Tr^X_{A,B}(f) \circ h$$

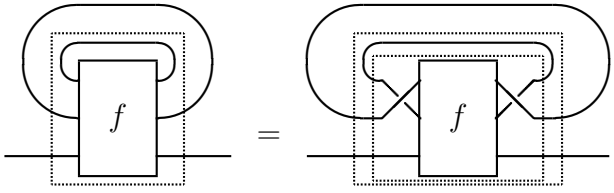


$$Tr^X_{X,X}(c_{X,X}) \circ \theta^{-1}_X = 1_X = Tr^X_{X,X}(c^{-1}_{X,X}) \circ \theta_X$$

$$Tr^X_{C\otimes A, C\otimes B}(1_C \otimes f) = 1_C \otimes Tr^X_{A,B}(f)$$



$$Tr^X_{A,B}(Tr^Y_{A\otimes X, B\otimes X}(f)) = Tr^Y_{A,B}(Tr^X_{A\otimes Y, B\otimes Y}((1_B \otimes c_{Y,X}) \circ f \circ (1_A \otimes c^{-1}_{Y,X})))$$



トレース演算子を持つバランス付きモノイダル圏を，**トレース付きモノイダル圏**（traced monoidal category）と呼ぶ．

トレース付きモノイダル圏の例として，第一にリボン圏が挙げられる．任意のリボン圏はトレース演算子をただひとつ持つ [19] （一意性については例えば [13] を参照）．有限次元線形空間の対称リボン圏 $\mathbf{Vect}^{\mathbf{fin}}_K$ では，トレース演算子は通常のトレースを対象 A, B でパラメータ化したものである．通常のトレースは $A = B = I$ の場合，つまり $Tr^X_{I,I} : \mathbf{C}(X, X) \to \mathbf{C}(I, I)$ に相当する．量子群の表

現のなすリボン圏では，量子トレースと呼ばれるもの（のパラメータ化）に他ならない．リボン圏でないトレース付きモノイダル圏の例としては，**Tangle** の対象を $+$ の列だけに制限した充満部分圏 $\mathbf{Tangle}_+$ がある．一般に，リボン圏の充満部分モノイダル圏はトレース付きモノイダル圏であるが，後述する構造定理より，実はその逆も成り立つ．

トレース付きモノイダル圏は，おそらく数学よりも計算機科学における認知度のほうがはるかに高いと思われる．重要な応用も，これまでのところほとんどが計算機科学におけるものである．以下では，その代表的な例である，再帰プログラムの意味論と相互作用の幾何について紹介する．

5. 再帰プログラムの不動点意味論

プログラム意味論における古典的な話題である，**再帰プログラム**（recursive program）の表示的意味論（**不動点意味論**）[35, 12] について，簡単な例をもとに説明しよう． 整数を入力に取り整数を出力する簡単なプログラム

$$\mathsf{sum}(x) \ \equiv \ \mathsf{if}\ x = 0\ \mathsf{then}\ 0\ \mathsf{else}\ x + \mathsf{sum}(x-1)$$

を考える．これは，プログラムの定義（$\equiv$ の右辺）で自身（sum）を呼び出す再帰プログラムになっている．たとえば，$\mathsf{sum}(3)$ の実行は，$3 \neq 0$ なので $3+\mathsf{sum}(2)$ と sum 自身を呼び出し，以下同様に続けて $3+(2+(1+\mathsf{sum}(0)))$ に到達し，$\mathsf{sum}(0) = 0$ なので和 $3+(2+(1+0))$ を計算して 6 を答えとして返す．sum のような再帰プログラムは，その実行が常に停止するとは限らないので，整数上の部分関数を定める．そして，sum の定める部分関数は，部分関数全体の上の（汎）関数

$$F(f)(x) \ = \ \begin{cases} 0 & (x{=}0) \\ x{+}f(x{-}1) & (x{\neq}0 \text{かつ} f(x-1) \text{が定義されている場合}) \\ \text{未定義} & (\text{その他の場合}) \end{cases}$$

の不動点として理解できる（$F(\mathsf{sum})\,(x) = \mathsf{sum}(x)$ が成り立つことを確認されたい）．sum に限らず，再帰プログラムの表示的意味は，適切な写像の不動点として与えることができる．領域理論はその代表的なものである．圏論を用いた表示的意味論では，そのような再帰プログラムの意味論に必要な不動点を取る操作は，以下のように定式化される．直積を持つ圏 $\mathbf{C}$ において、以下の条件をみたす関数の族 $(-)^\dagger : \mathbf{C}(A \times X, X) \to \mathbf{C}(A, X)$ を **Conway 不動点演算子**と呼ぶ：

- $f : A \times X \to X$, $h : A' \to A$ について $f^\dagger \circ h = (f \circ (h \times 1_X))^\dagger : A' \to X$
- $f : A \times Y \to X$, $g : A \times X \to Y$ について $(f \circ \langle \pi_{A,X}, g \rangle)^\dagger = f \circ \langle 1_A, (g \circ \langle \pi_{A,Y}, f \rangle)^\dagger \rangle : A \to X$
- $f : A \times X \times X \to X$ について $f^{\dagger\dagger} = (f \circ (1_A \times \Delta_X))^\dagger : A \to X$

これらの条件は，従来の再帰プログラムの不動点意味論のほとんどすべてについて成立している．特に，領域理論で用いられるカルテジアン閉圏はまさにそのような例になっている．

実は，Conway 不動点演算子は，トレース演算子の特別な場合であると言って良い．有限直積を持つ圏は，直積をテンソル積とみなすことにより，対称モノイダル圏の例になっている．有限直積をテンソル積とみなした対称モノイダル圏においては，Conway 不動点演算子とトレース演算子の間には一対一対応がつくことが，1990 年代中頃に，筆者と Martin Hyland によって独立に示された [11]．このことから，領域理論をはじめとするプログラム意味論で用いられている多くの圏が，実はトレース付きモノイダル圏になっていることがわかる [3]．結び目の理論で必要とされる非対称なリボン圏ではないけれども，少なくとも対称リボン圏の断片が，プログラム意味論において古くから用いられていたのである [4]．

この結果はさらに一般化でき，直積でないテンソル積を持つモノイダル圏においても，適当な（プログラム意味論でよく用いられるような）条件を満たすものにおいては，トレースから不動点演算子を構成することがわかっている．それを用いて，古典的なプログラム意味論では説明できなかった，巡回的な共有データ構造から生み出される再帰計算の意味論を展開することが可能である [11, 12, 13].

6. トレース付きモノイダル圏の構造定理と相互作用の幾何

トレース付きモノイダル圏のもうひとつの特徴付けのもととなる構造定理 [19] は，以下のようなものである．

> 任意のトレース付きモノイダル圏 $\mathbf{C}$ について，$\mathbf{C}$ からの，トレース付きモノイダル圏の構造を保つ充満忠実関手があるようなリボン圏 $\mathbf{Int}\,\mathbf{C}$ が存在する．しかも，この **Int** 構成 [5] は，リボン圏全体からなる 2-圏からトレース付きモノイダル圏全体のなす 2-圏への包含 2-関手の左双随伴（left biadjoint）を与える．[6]

ここで登場する **Int** 構成は，計算機科学において思いがけない応用をもたらした．それは，**Int** 構成により得られるリボン圏が，双方向の情報のやりとりを伴なう計算（相互作用）の意味論を与えるために非常に便利なものだからである．

トレース付きモノイダル圏 $\mathbf{C}$ に対し，リボン圏 $\mathbf{Int}\,\mathbf{C}$ は，$\mathbf{C}$ の対象の組を対象とし，$\mathbf{Int}\,\mathbf{C}$ における (X,U) から (Y,V) への射は，$\mathbf{C}$ における $X\otimes V$ から $Y\otimes U$ への射として与えられる．射の定義の中で，対象の第二成分の順番がひっくり返っている（反変になっている）ことに注意されたい．$\mathbf{Int}\,\mathbf{C}$ における射 $f:(X,U)\to(Y,V)$ と $g:(Y,V)\to(Z,W)$ の合成は，以下のように行う．

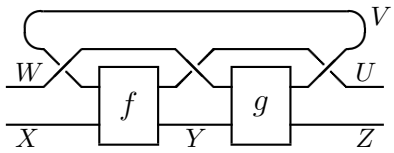
---

[3] 皮肉なことに，公理的領域理論では，Freyd の提案した（まっとうな数学から見て非常識な）条件から，Conway 不動点演算子，したがってトレース演算子の存在を導くことができる．

[4] トレース付きモノイダル圏の登場以前から，計算機科学の中でも巡回的・再帰的な構造をモノイダル圏を用いて代数的に取り扱う独自の試みが存在したことを付記しておく．特に 1980 年代の Ştefănescu の研究 [31] は記憶されるべきものである．

[5] **Int** 構成という名称は，整数の集合を，自然数の組の集合を適切な同値関係で割って構成することとの類似に由来する．

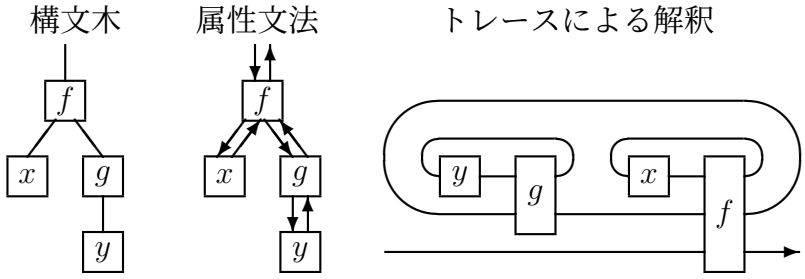
[6] 実は原論文 [19] ではトレース付きモノイダル圏全体のなす 2-圏の定義に小さな不備があり，後半の双随伴性に関する主張は正しくない．この問題は [15] で指摘・修正された．



左から右への素直な流れと，トレースを用いて右から左へと逆順に合成される流れとが共存していることがおわかりいただけるだろうか．計算機科学の感覚では，これはまさに「上り」と「下り」の情報流が共存する状況を表している．

Int 構成は，Girard が 1980 年代末に創始した双方向の情報流を伴なう計算の理論である**相互作用の幾何**（Geometry of Interaction）[10] の核心部分に暗黙のうちにあらわれている．のちに，Abramsky らは，**Int** 構成を基礎とした相互作用の幾何の再構成と一般化 [1] を与えた．

さらに，最近になって，勝股 [21] は，40 年以上前に Knuth が導入した，**属性文法**（attribute grammar）[24] と呼ばれるコンパイラ等で文脈自由言語の意味の記述に用いられる技法が，**Int** 構成を用いて明快に理解できることを示した．属性文法ではプログラムの構文木の構造に沿って双方向の情報のやりとりがなされるが，ここでもトレース付きモノイダル圏と **Int** 構成が重要な役割を果たすのである（下図）．



特に，高階のデータ（プログラム）を属性に用いる属性文法は、トレース付きモノイダル閉圏（traced monoidal closed category）上の **Int** 構成により理解できる．トレース付きモノイダル閉圏 **C** においては，**Int C** への埋め込みが右随伴関手を持ち [13]，このことから，高階属性を用いて双方向の情報流が一方向の情報流に帰着されることを説明できる．また，プログラムのコンパイル時の最適化技法であるプログラム変換においても，関連する結果が得られている [22].

7. 展望，あるいは願望

量子トポロジーから生まれてきた圏論的な構造が，プログラム意味論において重要な役割を果たしつつあることを紹介してきた．しかし，これで話はおしまい，ではない．確かにそれらの構造（トレース付きモノイダル圏，リボン圏）の具体例がプログラム意味論において用いられていることはわかった．しかし，それらはみな量子不変量で登場する具体例とはかなりかけ離れた存在であり，量子トポロジーの成果の本当の意味での応用とは言いがたいのではないか？それに，これまで見つかった例はいずれも対称なモノイダル圏であり，非自明な結び目の不変量にはなりえないから，結び目の理論の見地からは興味の対象外ではないか？

これらの批判は，実にもっともなものである．筆者としては，今後，具体的な例に踏み込んだ，プログラム意味論と量子トポロジーの一致点を見出したいと考えている．そのために，最近，量子トポロジーの専門家からは笑われそうなくらいに初歩的ではあるが，プログラム意味論で実際に用いられている圏において，量子不変量の類似物の構成を試みている [14]．出発点として，集合を対象とし，集合間の二項関係を射とする圏**Rel**を考える．**Rel**は，非決定的な計算のモデルとして，まだ線形論理のモデル等として，幅広い文脈で用いられている．**Rel**自身はコンパクト閉圏（対称リボン圏）の構造を持つ．いま，任意の群Gを考える．すると，**Rel**の中にGに対応する（通常の群環に相当する）Hopf代数が存在することがわかる．このHopf代数の量子二重化（quantum double）[5] を取ることにより，**Rel**の中でリボンHopf代数を構成でき，さらにその**Rel**における表現の圏を考えることができる．結果として得られるのは，交差G集合（crossed G-set）[8] を対象とし，交差G集合の構造を保つ二項関係を射とするリボン圏である．

この例自体は，教科書にも載っている通常の量子二重化をそのまま真似したものであり，得られる不変量にも，あまり目新しさはない．しかし，筆者の知る限り，これは，プログラム意味論の文脈で得られた最初の非対称なリボン圏である．再帰プログラムの意味論と組み合わせることも可能な，最初の例である．全然見当違いである可能性もあるが，ここから，プログラム意味論の量子化，という夢みたいな話をはじめてみるのも悪くないのではないだろうか．リボン圏まで来れば，モジュラー圏（modular tensor category）[33, 3]，そして位相的量子場の理論と位相的量子計算 [6] の世界へも，あと一息である．プログラム意味論の手の届く範囲が，意外な方向で大きく拡張されるかも知れない．少なくとも，プログラム意味論と量子トポロジーとの間で，より実りあるテクノロジー・トランスファーがなされるようにしていきたい，と，Freydの言葉に立ち返りつつ願うものである．

参考文献

[1] S. Abramsky, E. Haghverdi and P.J. Scott, Geometry of Interaction and linear combinatory algebras, Math. Struct. Comp. Sci. **12**(5) (2002), 625–665.

[2] S. Abramsky and A. Jung, Domain theory, Handbook of Logic in Computer Science Vol. 3, Oxford University Press (1994), 1–168.

[3] B. Bakalov and A. Kirilov, Lectures on Tensor Categories and Modular Functors, University Lecture Series, 21, American Mathematical Society, 2001.

[4] R. Crole, Categories for Types, Cambridge University Press, 1993.

[5] V.G. Drinfel'd, Quantum groups, In Proc. Intrn. Congr. Math., Berkley, 1986, vol. 1, pp. 798–820, 1987.

[6] M.H. Freedman, A. Kitaev and Z. Wang, Simulation of topological field theories by quantum computers, Commun. Math. Phys. **227** (2002), 587–603.

[7] P. Freyd, Algebraically complete categories, In Category Theory, Springer Lecture Notes in Math. **1488**, pp. 95–104, 1990.

[8] P. Freyd and D.N. Yetter, Braided compact closed categories with applications to low dimensional topology, Adv. Math. **77** (1989), 156–182.

[9] J.-Y. Girard, Linear logic, Theoret. Comp. Sci. **50** (1987), 1–102.

[10] J.-Y. Girard, Geometry of Interaction I: interpretation of system F, In Proc. Logic Colloquium '88, pp. 221–260, 1989.

[11] M. Hasegawa, Models of Sharing Graphs: A Categorical Semantics of let and letrec, Distinguished Dissertations Series, Springer-Verlag, 1999.

[12] 長谷川真人, 再帰プログラムの意味論について, 数学 **59** (2007), 180–191.

[13] M. Hasegawa, On traced monoidal closed categories, Math. Struct. Comput. Sci. **19** (2009), 217–244.

[14] M. Hasegawa, Bialgebras in Rel, In Proc. Mathematical Foundations of Program Semantics, Electron. Notes Theor. Comput. Sci. (2010).

[15] M. Hasegawa and S. Katsumata, A note on the biadjunction between 2-categories of traced monoidal categories and tortile monoidal categories. Math. Proc. Cambridge Phils. Soc., **148** (2010), 107–109.

[16] 神保道夫, 量子群とヤン・バクスター方程式, シュプリンガー・ジャパン, 1990.

[17] A. Joyal and R. Street, The geometry of tensor calculus, I, Adv. Math. **88** (1991), 55–113.

[18] A. Joyal and R. Street, Braided tensor categories, Adv. Math. **102** (1993), 20–78.

[19] A. Joyal, R. Street and D. Verity, Traced monoidal categories, Math. Proc. Cambridge Phils. Soc. **119** (1996), 447–468.

[20] C. Kassel, Quantum Groups, Grad. Texts in Math. **155**, Springer-Verlag, 1995.

[21] S. Katsumata, Attribute grammars and categorical semantics, In Proc. ICALP, Springer Lecture Notes in Comput. Sci. **5126** (2008), pp. 271–282.

[22] S. Katsumata and S. Nishimura, Algebraic fusion of functions with an accumulating parameter and its improvement, J. Funct. Programming **18** (2008), 781–819.

[23] G.M. Kelly and M.L. Laplaza, Coherence for compact closed categories, J. Pure Appl. Algebra **19** (1980), 193–213.

[24] D.E. Knuth, Semantics of context-free languages, Mathematical Systems Theory **2** (1968), 127–145.

[25] J. Lambek and P. Scott, Introduction to Higher-order Categorical Logic, Cambridge University Press, 1986.

[26] S. Mac Lane, Categories for the Working Mathematician, Grad. Texts in Math. **5**, Springer-Verlag, 1971.

[27] P.-A. Melliès, Functorial boxes in string diagrams, In Proc. Computer Science Logic, Springer Lecture Notes in Comput. Sci. **4207** (2006), pp. 1–30.

[28] T. Ohtsuki, Quantum Invariants, A Study of Knots, 3-Manifolds, and Their Sets, World Scientific, 2002.

[29] D.S. Scott, Data types as lattices, SIAM J. Comput. **5** (1976), 522–587.

[30] M.-C. Shum, Tortile tensor categories, J. Pure Appl. Algebra **93** (1994), 57–110.

[31] G. Ştefănescu, Network Algebra, Springer-Verlag, 2000.

[32] 照井一成, 線形論理の誕生, 数学 **62** (2010), 115–132.

[33] V.G. Turaev, Quantum Invariants of Knots and 3-Manifolds, de Gruyter, 1994.

[34] D.N. Yetter, Functorial Knot Theory, World Scientific, 2001.

[35] 横内寛文, プログラム意味論, 共立出版, 1994.