チュートリアル

『ソフトウェアサイエンスの基本』シリーズ第 2 回

帰納と再帰 ―表示的意味論の第一歩―

木下 佳樹

ごく基本的な数学的リテラシーだけを前提として，集合の帰納的定義，函数の再帰的定義を説明する数学的構造を，会話形式で説明する．

The mathematical structure for inductive definition of sets and recursive definition of functions is presented through dialogue of Kuka and Racha. Only basic mathematical literacy is assumed.

苦迦と羅茶は，計算機科学の研究に従事する同僚である．小春日和の午後，珍しくセミナーにも会議にも出ずにすんだ二人は，どこへ向かうのかわからないおしゃべりを楽しんでいる．

羅茶 今さら言うのもなんですが，再帰呼び出しというのは，わかりきったようでいて，なんだかわからないところがありますね．

苦迦 プログラムでの手続きや函数[†1]の定義をするときに，自分の定義に自分自身を呼び出す場合があるという，いわゆる自己参照的 (self-referential) な定義の話ですね．定義されたものをどのように実行したり計算したりするか，つまり，手続きのなかで自分自身を呼び出す場合にどのようにコンパイルするかは，羅茶さんのことだから，百もご承知なんでしょう．自己参照的に定義された手続きや函数が，どんなものなのかということになると，どうもハッキリしない，ということでしょうか．

羅茶 うーん，たしかに，その自己参照的定義ですか，そういうふうに定義された函数の計算の仕方ははっきり述べられているわけですから，それ以上何を求めるのだ，という気もするのですよ．どこがわからない気がするのかな…….

苦迦 自己参照的な定義で定義されるものが何なのかはわかるが，そもそも自己参照的な定義とは何なのか，と言うことのイメージがつかめないということかもしれませんね．

羅茶 そういうところが確かにありますね．定義された函数が何なのかは，一応分かった気がしていますが，そのときに使った定義の仕方が，ひょっとしたら循環論法なんじゃないかと疑いたくなるようなものですから．

苦迦 定義とは何か，なんて，普通は考えなくても自明なものにみえますが，自己参照的な定義は，ちょっとややこしくて，自明ではありませんからね．

羅茶 自己参照的定義とはこういうものだ，と言うような説明がほしいのかな．

再帰呼び出しは，プログラミングの基本的な技法である．50 年前ならともかく，今ではプログラミングの入門コースで学ぶ概念であろう．現役の研究者である羅茶ともあろうものが，わかったようでわから

Basics of Software Science 2: Induction and Recursion—A First Step to Denotational Semantics.
Yoshiki Kinoshita, 産業技術総合研究所情報技術研究部門, Research Institute for Information Technology, AIST.
コンピュータソフトウェア, Vol.29, No.1 (2012), pp.30–46.
[解説論文] 2011 年 8 月 15 日受付.

†1 現在の我が国の中等教育向け教科書では「関数」と記されるが,「函數」がもともとの表記であり，これは音訳でもある.「函」が常用漢字にないという理由で，折角の音訳を捨てるには忍びない.「數」を「数」と表記することに反対するつもりはないので，本稿では「函数」と記す.

ない，というのはいったいどういうことなのだろうか．再帰呼び出しで定義された函数がどのようなものであるのかを明らかにしたい，ということであれば，何でもかんでも集合として表現する，という 20 世紀数学の方針に従いたい，ということのようにもきこえる．だとすれば，これはいわゆる表示的意味論 (denotational semantics) の場を提供する，ということにつながるのかもしれない．しばらく彼らの閑談を横で聞いてみよう．

1 集合の帰納的定義

苦迦 自己参照的定義をある種の「変換」によって表わす理論があります．

羅茶 自己参照的定義を変換で表わすというわけですか．それができれば話はすっきりしますね．

苦迦 自己参照的に定義されるのはプログラムの手続きや函数だけではありません．データ型，つまり集合を自己参照的に定義することも現代的なプログラミング言語では行われています．集合の自己参照的な定義を帰納的定義 (inductive definition) といい，函数の自己参照的な定義を再帰的定義 (recursive definition) といって帰納と再帰を使い分ける場合もあるようです．もっともそのような区別をしない場合も多いので，函数の帰納的定義，なんていう言葉をきいても驚くには及びません．

羅茶 そういえば函数型言語では，函数だけじゃなくて，データ型をも自己参照的に定義する機能が用意されていますね．

苦迦 図 1 では，いくつかの言語で二分木のデータ型 Tree を定義していますが，いずれも，Tree と言う集合を定義する次の三箇条の日本語文に相当するものです．

規則 1. empty は Tree の要素である．

規則 2. x と y が Tree の要素であれば，cons x y も Tree の要素である．

規則 3. 以上の規則を有限回繰り返して Tree の要素であると確認できるものだけが Tree の要素である．

規則 2 では Tree によって Tree を定義しているように見えます．あたかも循環論法に陥っているかのようなのに，実はうまく集合が定義される，という仕掛けです．

Haskell による定義

```
data Tree = empty | cons Tree Tree
```

Standard ML による定義

```
datatype Tree =
    empty | cons of Tree * Tree
```

Agda による定義

```
data Tree :  Set where
    empty :  Tree
    cons  :  Tree → Tree → Tree
```

図 1 二分木のデータ型

羅茶 こういった自己参照的な定義を変換で表わす，とのことでしたが，どんな筋書きなのか，ざっと話していただけませんか．

苦迦 今の場合は集合を定義しようとしているので，集合を別の集合にかえる変換を考えます．さて，規則 1，2，3 は，Tree という集合を定義するためのものです．この規則をよく見てみると，既に Tree の要素である，とわかっているものをもとにして，新たに Tree の要素であるとわかるものが生じる，という形をしています．規則 2 では x，y の二つが，既に Tree の要素であるとわかっているもの，そうして cons x y が，この規則によって新たに Tree の要素であるとわかるものです．

羅茶 はい，Tree の要素をつかって Tree の要素を定義しているわけで，そこがまさに自己参照的なところですね．

苦迦 規則 1 ではたまたま，「既に Tree の要素であるとわかっているもの」を前提とすることなく，empty が Tree の要素である，といっています．こういうのもいいわけです．右側のプログラミング言語での定義では，“ = ”の左辺に現れるかと右辺に現れるかで，「既に Tree の要素であるとわかっているもの」なのか「新たに Tree の要素で

あるとわかるもの」なのかを区別することができます.

羅茶 なるほど，規則 1 と 2 は,「既に Tree の要素である，とわかっているものをもとにして，新たに Tree の要素であるとわかるものが生じる」という形をしていますが，規則 3 は少し違うようですね．Tree の要素が新たに生じる，というのではなく，これ以外の方法では要素であるとわかるものはない，といっているのですから.

苦迦 そうです．実は，規則 3 は帰納的定義の締めくくりに必ず出てくる決まり文句です．closing sentence などと呼ばれています．決まり文句なので，省略される場合も多いのです.

羅茶 確かに，函数型プログラミング言語による記述には，規則 3 に相当する部分がみられませんね.

苦迦 はい．closing sentence は，ここで定義される集合の，ある意味での最小性を主張しています.

羅茶 どういう意味での最小性でしょうか.

苦迦 つまり，ほかの規則たち，この場合だと規則 1 と 2 を満足するような集合は一つに決まるとは限らないのですよ．そういうものの中で最小のものこそが，ここで定義しようとしている集合なのだ，と定めるのが規則 3 のような closing sentence です.

羅茶 なるほど．closing sentence は省略されることもある，とのことでしたが，重要な意味を持っているのですね．さて，これらの規則を，どのように読むと，集合の定義であると解釈できるのでしょうか.

苦迦 「既に Tree の要素であるとわかっているものの集合」を X とし,「X をもとにして Tree の要素であるとわかるものの集合」を $F(X)$ としましょう．これらの規則は，$F(X)$ がどんな集合であるのかを示していると見なすことができます.

羅茶 $F(S)$ はどんな集合でしょうか?

苦迦 $F(S) = \mathbf{1} + S \times S$ です.

羅茶 なんだ，結局足し算と掛け算か.

苦迦 しかし，いうまでもありませんが，これらは数の加算や乗算ではありません．集合への操作です．$+$ は直和，$\times$ は直積です.

$$A + B = \{\,(0,a)\,|\,a \in A\,\} \cup \{\,(1,b)\,|\,b \in B\,\}$$
$$A \times B = \{\,(a,b)\,|\,a \in A, b \in B\,\}$$

図 2　直和と直積

$$\mathbf{1} = \{\,*\,\}$$

図 3　一点集合

羅茶 えーと直積 $A \times B$ は A の要素と B の要素の対の集合でしたね．直和ってなんでしたっけ?

苦迦 直和 $A + B$ は A と B の合併集合のようなものですが，単なる合併集合ではなく，各要素にタグをつけて，その出自，つまり A からきたものなのか B からきたものなのかが明確であるようなものです．式で書くと，図 2 のようになります.

羅茶 あ，なるほど，A にも B にも属する要素がある場合，普通の合併集合だとどちらからきた要素と見るべきなのか，はっきりしませんからね．このように 0 や 1 で目印をつけておくと出自がはっきりするというわけか．目印は 0 と 1 でなくとも構いませんね.

苦迦 それはそのとおりで，$*$ と @ を目印にしても構いません．これについてはいろいろいいたいこともありますが，どんどん横道にそれるので，今は我慢しておきましょう.

羅茶 えーと，F の定義に表われる $\mathbf{1}$ は数の 1 ですか?

苦迦 いえ，唯一つのものからなる集合です．英語で singleton と呼ばれています．日本語だと一点集合 (図 3) かな.

羅茶 ふーむ，その唯一つのものとは何でしょう?

苦迦 実のところ何でもいいんです．さしあたって $*$ とでも記しておくことにしましょうか.

羅茶 とはまたいい加減な.

苦迦 いい加減に見える取り扱いをする理由があるのですよ．つまり何を持ってきても，集合としては同型ですから.

どこへ向かうのかわからぬ話をしているように見

えても，当人たちには一定の方向感覚があるようで，やたら横道に逸れることを気にしているようだ．野次馬にとって横道に逸れてもらっても一向に構わない．それどころか，逸れたほうが面白いかもしれないのだが……．

羅茶 なるほど．さっきの函数型言語での例だと，その要素を empty と呼ぶことになりますか．そうすると，$S \times S$ のところは，cons にあたるわけですね．

苦迦 まあそうです．「cons にあたる」というのがどういう意味なのか，が明らかだとしての話ですがね．ここで，木の要素を右辺で持ってこなければならないときには S の要素としているのに対して，左辺ではそれが $F(S)$ になっていることに注意してください．

羅茶 待てよ，定義が F のような変換で表わされる，というのでしたが，F によって定義される集合とはどんなものでしょう．

苦迦 一段ずつ，段階をふんで考えます．F は集合を別の集合に写す変換ですが，その意図は，「既に Tree の要素だと分かっているものの集合」を，「それをもとに Tree だと分かるものの集合」に写すことでした．第 n 段階で Tree の要素だと分かっているものの集合を $T(n)$ としましょう．はじめはまだ，Tree の要素だと分かっているものは何もないので，空集合 $\varnothing$ から始めます．

羅茶 すると $T(0) = \varnothing$ ですね．

苦迦 そのとおりです．第 $n+1$ 段階では，第 n 段階で Tree の要素だと分かっているものの集合に F を施して，Tree の要素だと分かるものを増やします．必ず増えることが証明できるわけではありませんが，我々はそう期待したいわけです．

羅茶 そりゃそうだ．$T(n+1) = F(T(n))$ というわけですね．すると，数学的帰納法をつかって，一般に $T(n) = F^n(\varnothing)$ といえますね．

苦迦 さてここで，F は単調であること，つまり $X \subseteq Y$ であれば $F(X) \subseteq F(Y)$ であることに注意します．これがいえれば，我々の期待通り $F^n(\varnothing) \subseteq F^{n+1}(\varnothing)$ がいえます．

羅茶 本当かな．まず F が単調かどうか．$F(X) = \mathbf{1} + X \times X$ だったから $X \subseteq Y$ を仮定して $\mathbf{1} + X \times X \subseteq \mathbf{1} + Y \times Y$ がいえればいい．$\mathbf{1} + X \times X$ の要素がすべて $\mathbf{1} + Y \times Y$ に属することを確かめよう．$\mathbf{1} + X \times X$ の要素は，$(0, *)$ か $(1, (x_1, x_2))$ の形です．ここで x_1 と x_2 は X の要素であればなんでもよいわけだ．さて，$(0, *)$ はもちろん $\mathbf{1} + Y \times Y$ の要素です．これは $+$ の定義からすぐにわかります．では $(1, (x_1, x_2))$ の形の要素はどうか．$x_1, x_2 \in X$ ですが，$X \subseteq Y$ と仮定しましたから，$x_1, x_2 \in Y$ でもある．そうすると $(x_1, x_2) \in Y \times Y$ がわかるから，$(1, (x_1, x_2)) \in \mathbf{1} + Y \times Y$ を結論することができます．

苦迦 これで F が単調であることが確かめられました．

羅茶 このことから $F^n(\varnothing) \subseteq F^{n+1}(\varnothing)$ がいえるのかな．n に関する数学的帰納法で確かめてみましょう．まず $n = 0$ の場合: $F^0(\varnothing) \subseteq F^1(\varnothing)$ かどうか．左辺は $\varnothing$ に等しいですね．あ，こりゃ明らかだ．$\varnothing$ はどんな集合にも含まれますから $\varnothing \subseteq F^1(\varnothing)$．次に $F^n(\varnothing) \subseteq F^{n+1}(\varnothing)$ を仮定して $F^{n+1}(\varnothing) \subseteq F^{n+2}(\varnothing)$ を示してみましょう．あ，ここに F が単調である，という前提を使うことができますね．つまり，$F^n(\varnothing) \subseteq F^{n+1}(\varnothing)$ であれば，前提から $F(F^n(\varnothing)) \subseteq F(F^{n+1}(\varnothing))$，つまり $F^{n+1}(\varnothing) \subseteq F^{n+2}(\varnothing)$ というわけです．

苦迦 はい．これで任意の n について $F^n(\varnothing) \subseteq F^{n+1}(\varnothing)$ であることが確かめられました．この結果，$F^0(\varnothing) \subseteq F^1(\varnothing) \subseteq F^2(\varnothing) \subseteq \ldots \subseteq F^n(\varnothing) \subseteq F^{n+1}(\varnothing) \subseteq \ldots$ がわかります．

羅茶 つまり，集合の列 $(F^n(\varnothing) \mid n \in \mathbb{N})$ は $\subseteq$ に関して昇列だというわけですね．

苦迦 図を書くとすると，図 4 のようになるでしょうか．さて，この昇列の合併集合 $L = \bigcup \{ F^n(\varnothing) \mid n \in \mathbb{N} \}$ を考えましょう．

羅茶 ああ，$F^n(\varnothing)$ は，第 n 段階で Tree の要素だと分かっているものの全体ですから，それらの合併集合をとると，「そのうちに Tree の要素だと分かるものの集合」ということになります．それがすなわち Tree という集合だ，と決めてしまえばよ

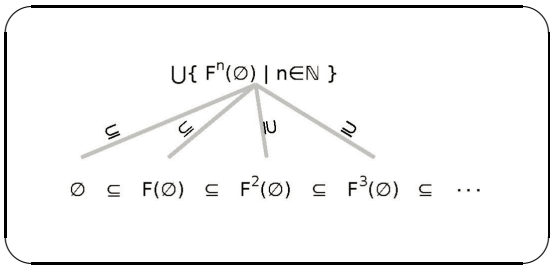

図 4 $X = F(X)$ **の解の逐次的近似**

いですね.

苦迦 はい. そうして, 実は L は F の不動点になっています.

羅茶 へえ, つまり集合 L とそれに変換 F を施して得られる集合 $F(L)$ が等しいということですね. L に対してはもはや, F を施しても新たに Tree と分かる要素はない, ということか.

苦迦 $L \subseteq F(L)$ を確かめてみましょう. $x \in L$ とすると, $L = \bigcup\{F^n(\varnothing) \mid n \in \mathbb{N}\}$ ですから, $x \in F^k(\varnothing)$ となる k があります. 既に見たように $F^k(\varnothing) \subseteq F(F^k(\varnothing))$ ですから $x \in F(F^k(\varnothing))$, しかし $F^k(\varnothing) \subseteq L$ ですから, F の単調性によって $x \in F(L)$ がわかります.

羅茶 なるほど. 逆に $F(L) \subseteq L$ もいえるかな. $y \in F(L)$ として $y \in L$ を示してみましょう. $F(L) = \mathbf{1} + L \times L$ ですから, y は $(0, *)$ か $(1, (u, v))$ の形をしています. ここで u, v は L の要素ですね. $y = (0, *)$ のときは, これはたとえば $F(\varnothing)$ の要素ですが, $F(\varnothing) \subseteq L$ ですから $y \in L$ ですね.

苦迦 $F(\varnothing)$ に限らず, $n \geq 1$ であれば y は $F^n(\varnothing)$ の要素ですね.

羅茶 $y = (1, (u, v))$ の形の場合, $u, v \in L$ なのですから, それぞれ $u \in F^h(\varnothing)$, $v \in F^k(\varnothing)$ となる h, k があるはずです. h, k のうち大きいほうを j とすれば, $F^n(\varnothing)$ は単調増加であり, u, v はともに $F^j(\varnothing)$ の要素ですから, $(1, (u, v)) \in 1 + F^j(\varnothing) \times F^j(\varnothing) = F^{j+1}(\varnothing)$. しかし, $F^{j+1}(\varnothing) \subseteq L$ でしたから, 結局 $y = (1, (u, v)) \in L$ ということになります.

苦迦 L が F の不動点であるということが確かめられましたね. ところで, F の不動点は L だけじゃなくて, 二つも三つもあるかも知れません.

羅茶 そういわれれば, 確かに一つとは限らない. もっとも二つ目を出してみろと言われても, すぐには出せませんが, といって, 一つしかないんだと言うことを証明せよといわれても困るなあ…….

苦迦 実際に, F の不動点は一つではないのですよ. L の特別なところは, F の不動点たちのなかで, L は最小のものであるということです.

羅茶 最小? あ, つまり集合として小さい, $\subseteq$ について最小, ということなんですね. そんなことが証明できますか.

苦迦 はい. 実は我々が今議論してきたようなことは,「Kleene の不動点定理」という形で整理されています.

羅茶 数学の話になるわけですね. それは結構ですが, そうすると, はじめに挙げられた規則 1–3 によって定義される集合は, F の最小不動点であって, 単なる不動点ではないということになりますか.

苦迦 そのとおりです. さっきもお話ししたように, 規則 3 がまさに, 定義される集合が, 3 以外の規則を満たすもののなかで最小のものであることを要求していますよね.

羅茶 なるほど. 規則 1 と 2 が不動点であることを要求していて, 規則 3 が最小不動点であることを要求している, ということですね.

苦迦 はい. 今までの議論を振り返ってみると, 集合 Tree を自己参照的に定義する日本語文を表わす集合の変換 F を抽出しました.

羅茶 ああ, これが話を数学化する第一歩だったのだな.

苦迦 そうして, F の最小不動点を構成して, それを F によって定義される集合としたわけです.

羅茶 自己参照的な定義の正体は集合の変換なのだ, というわけですね.

苦迦 ところで, Tree の他にもいろいろな自己参照的な定義はありえますが, 今の議論を適用するためには, 定義を表わす集合の変換 F が単調であること, つまり $X \subseteq Y$ ならば $F(X) \subseteq F(Y)$

であることが必要です.

羅茶 確かに，このことを我々は証明しましたね.

苦迦 実はもう一つ F に関する条件を使っているのですが，詳しいことはともかく，応用上大事なのは，F が直和と直積を組み合わせて作ったものであれば，それらの条件を満足し，我々の議論を適用できる，ということです.

羅茶 $+$ と $\times$ の組み合わせはOK，というわけですね.

苦迦 はい．$+$ と $\times$ を組み合わせてできる集合の変換は，多項式函手 (polynomial functor) と呼ばれています．函手というのは圏論 (category theory) の言葉で，集合の変換に使うのは変に聞こえるかもしれませんが，圏論の意味での函手に簡単に拡張できます.

羅茶 多項式と呼んでしまうと覚えやすいですね.

苦迦 F のかわりに多項式函手であるような変換を持ってきても，今の議論と同じようにして，最小不動点を構成することができます.

羅茶 そのようなことができる自己参照的定義は，循環論法に陥ることがないまっとうなもの，ということができそうですね．まっとうでない，つまり最小不動点を持たなかったり，不動点すら持たない変換もありえますか.

苦迦 あります．また，つまらない不動点しかない変換もあります．例えば集合 A を集合 $[A, A]$ つまり A からそれ自身への写像の全体が作る集合にうつす変換の不動点は一点集合しかありません．要素の個数を勘定すると，そういうことになってしまいます．A の要素の個数を $|A|$ とかくと，$|[A, A]| = |A|^{|A|}$ ですが，$x = x^x$ の解となる自然数は 1 しかありませんから．つまり，不動点があったとしても一点集合でしかありえないというわけです.

羅茶 ええっ！ $[A, A]$ はいわゆる函数型じゃああありませんか．だって，函数プログラミングでは，こういうものもどんどん使っていますよ.

苦迦 集合の変換としては不動点を持たない，したがって，今の我々の説明のしかたは通用しない，と言うだけのことです．だからいわゆる函数型を説明するには，集合の変換ではない道具立てが必要です．その理論は，創始者の Dana Scott にちなんで Scott 理論と呼ばれることもあります.

羅茶 ははあ．すると，今回の議論は，直和と直積だけで，函数型がはいらないようなデータ型の定義には，少なくとも使えるのだ，と思っていたらいいですね.

苦迦 そうです．例えば集合 A が既に与えられているときに，A の要素のリストの集合は $\mathrm{List}_A(X) = \mathbf{1} + A \times X$ という変換で表わされるし，A の要素をアトムとする Lisp の S 式は $\mathrm{Sexp}_A(X) = \mathbf{1} + A + X \times X$ という変換で表わされます.

羅茶 おっと，一息におっしゃいましたが，この変換ひとつずつについて，今までの議論と同じくらいの検討が必要ですね.

苦迦 演習問題ということになるでしょうか.

羅茶 ところで別に最小でなくとも，最大の不動点でもいいですね．存在すれば，の話ですが．つまり，集合の変換が定義する集合を，その変換の最大不動点にするのでもよさそうです．あ，両方あったらどうしよう．三つ以上不動点があったらもっと迷いますね.

苦迦 決め事ですから，どれかに決めればよいのですが，実際，最大不動点を考えたほうが具合のいい場合もあります.

羅茶 そうでしたか.

苦迦 自己再帰的な定義が表わす写像の最小不動点をその定義が定めるものとみなす場合に，再帰的定義 (recursive definition) (函数の場合)，帰納的定義 (inductive definition) (集合の場合) とよび，最大不動点をとる場合には，余再帰的定義 (corecursive definition)，余帰納的定義 (coinductive definition) と呼ぶことも，広く行われています.

羅茶 何ですか，その「余」というのは?

苦迦 co- という双対を表わす接頭辞に当てている漢字のようです．ほら，三角関数で正弦，余弦というでしょう.

羅茶 ああ，sine が正弦，cosine が余弦ですね．でも余帰納的定義は最大不動点をとるということでした．cosine のように，sine から $\pi/2$ ずらす,

というような操作をしているわけではありませんが…….

苦迦 まあ，ここで「余」といっているのは，対というか，組になっているということです.

羅茶 なるほど.

苦迦 ちなみに，最小不動点，最大不動点などといっているので，順序を考えることが集合の帰納的定義のために必須であるように見えるかもしれませんが，そうでもありません．順序を導入する代わりに，ある種の位相空間を導入して，唯一つの不動点が存在する，と言うような状況を作る理論もあります．もっとも，順序集合は位相空間と密接に関係するので,「順序の代わりに位相を入れる」というのはちょっとへんな言い方ではありますが.

羅茶 なんだかよくわからなくなってきましたが，とにかく，いろいろな理論が立てられるわけで，今回のお話が絶対的で唯一のものであるというわけではないのですね．集合の帰納的定義を集合の変換で表現する理論について，大体の筋道を話してくださったのですが，函数の再帰的定義はどのようにとらえることができるでしょうか.

2 函数の再帰的定義

苦迦 集合の帰納的定義と同じように，函数の再帰的定義も，函数の変換によって表現することができます.

羅茶 どんな変換なのかな．集合を定義するのに集合の変換を使ったところをみると，函数の定義のためには，函数の変換をつかうのでしょうか?

苦迦 そうしたいところですが，それでは少し具合が悪い．集合の間には包含関係 ($\subseteq$) があって，さきほどはそれをうまく使ったわけですが，ふつうの函数の間にはそういう半順序関係がありません.

羅茶 そういえばそうですね．人工的に順序を作りますか?

苦迦 まあ，いってみればそういうことなのですが，対象をふつうの函数だけでなく，部分函数にまで広げて，そこに順序をいれます．でも，自然に見える順序ですよ.

羅茶 ははあ．部分函数ですか．函数の一部分?

苦迦 いえ，そうではなくて，部分的にしか定義されていない函数，とでもいうべきでしょうか．f が集合 A から B への函数であれば，A のすべての要素 a に対して $f(a) \in B$ が定まっていますよね.

羅茶 はい．全域性というのでしたっけ.

苦迦 そうです．これに対して，f が部分函数であれば，$f(a)$ が定義されない場合もあっていいのです.

羅茶 定義されない，というのはわかりにくいですね．定義されないということを定義してください，なんて，禅問答みたいになってしまった.

苦迦 A から B への部分函数は A から $B+\mathbf{1}$ への (全域的) 函数である，と定義することができます.

羅茶 えーと，$\mathbf{1}$ は $*$ のみを要素とする一点集合でしたから $B+\mathbf{1} = \{\,(0,b)\,|\,b \in B\,\} \cup \{\,(1,*)\,\}$ ですね．f が A からこの集合への函数というわけか．$f(a)$ の値は $(0,b)$ の形かあるいは $(1,*)$ ということになります．そうか，前者の場合は値が b である場合に相当し，後者の場合は値が定義されない場合に相当する，というわけですね.

苦迦 そのとおりです．集合 $B+\mathbf{1}$ を $B_\perp$ と書いて，B のリフト，と呼ぶこともあります.

羅茶 一つ要素を付け加えて持ち上げる，という感じなんでしょうかねえ.

苦迦 その場合，$(1,*)$ を $\perp$ あるいは $\perp_B$ などと書くことがあります．また，$(0,b)$ を単に b と略記することも多いですね．ところで，部分函数に対して，普通の函数のことを，全域的に定義されているという意味で，全域函数と呼ぶことにしましょう．全域函数と違って，部分函数の間には自然な半順序 $\sqsubseteq$ があります．つまり，$f \sqsubseteq g$ となるのは，任意の a に対して

1. $f(a)$ が定義されていない ($f(a) = \perp$) か，あるいは
2. $f(a)$ も $g(a)$ も共に定義されていて，それらが等しい ($f(a) = g(a) = (1,b)$) か

のいずれかだ，ということであると定めます.

羅茶 なるほど，$\sqsubseteq$ の意味で大きい部分函数は，定義される範囲がより広いが，既に定義されていると

$$n! = \textbf{if } n = 0 \textbf{ then } 1 \textbf{ else } (n+1) \times (n!)$$

図 5 階乗 $n!$ の再帰的定義

ころでは値は変わらない，ということになりますね．

苦迦 集合の帰納的定義では，単調な変換 F を利用して，定義される集合の候補を段階を追って大きくしていったわけですが，おなじように，函数の再帰的定義でも，部分函数の変換を考えて，定義される部分函数の候補を段階を追って大きくしていくことを考えます．

羅茶 そのために順序関係を使ったわけですね．待てよ，しかしそうすると，最後に定義できるものも一般に部分函数であって，全域的な函数とは限らない，ということになりますか?

苦迦 全くそのとおりです．函数の値の計算が無限ループに陥る場合が，値が定義されない場合に対応すると考えるとよいですね．

羅茶 階乗は図 5 のように定義することができますが，これを例にとって，部分函数のどのような変換を考えるのか，説明していただけませんか?

苦迦 なるほど，階乗の定義は再帰的定義の定番かもしれませんね．図 5 でも $+$ と $\times$ が使われていますが， Tree のときのように $+$ で集合の直和を表わしたり $\times$ で直積を表わしたりするわけではなく，ここでは数の加算と乗算を表わすのですよね．

羅茶 はい，そうです．何で，直和に加算と同じ $+$ を使うんでしょうかね．

苦迦 集合の直和の要素数は，各々の集合の要素数の和になっていますよ．直積と積だって同じことです．

羅茶 なるほど．全く無縁でもないんだな．

苦迦 さて，この定義を表わす変換は $G(f) = \lambda n.\ (\textbf{if } n = 0 \textbf{ then } 1 \textbf{ else } n \times f(n \dot{-} 1))$ のようなものです．

羅茶 右辺の “$\lambda n.$” というのは何でしょう．

苦迦 あ，これは λ 記法と呼ばれているもので，「この式は n を仮引数とする函数である」と言うことを表わしています．例えばこの式は，仮引数が n，本体が **if** ... であるような部分函数を表わします．つまり，自然数 n を if $n = 0$ then 1 else $n \times f(n \dot{-} 1)$ に写すような部分函数ですね．なんで λ を使うんだったかな，確か創始者の A. Church が亡くなったときにそのことについての電子メイルが飛交っていましたが…….

羅茶 なるほど．そうか，G は部分函数を部分函数に写すものでしたから，G による f の行き先も部分函数なんですね．だから，函数を式で表わす必要があるわけですね．

苦迦 そうなんです．私自身は

$$[n \mapsto \textbf{if } n = 0 \textbf{ then } 1 \textbf{ else } n \times f(n \dot{-} 1)]$$

のような記法が好きで，自分のメモではよくこういうものを使います．いずれにしろ，引数が何かを明確にする表記が必要です．式の中に二つ以上の変数が現れる場合もあって，その場合には，引数がどれなのかをハッキリさせる必要があるわけです．

羅茶 積分で $\mathrm{d}x$ なんて書くのと似ていますね．もうひとつ，$\dot{-}$ はなんですか?

苦迦 これは自然数の減算です．結果も自然数にしたいので，$m < n$ のときには $m \dot{-} n = 0$ と決めてしまいます．

羅茶 なるほど．あれれ，右辺を見ていると部分函数とはいっても全域函数に見えてきましたよ．

苦迦 たしかに，f が全域的に定義されておれば，右辺もそうですが，$f(n)$ の値が定義されていない場合には，$n \times f(n \dot{-} 1)$ の値も定義されません．

羅茶 なるほど．$h \times k$ の値は，h の値が定義されなかったり，あるいは k の値が定義されなかったりすると，定義されませんからね．例えば，と，極端な場合ですが，どの n についても値が定義されていないような部分函数を考えて見ましょう．何か記号がほしいですね．うーん，いつも $\perp_{\mathbb{N}}$ を値にする函数だからこれも $\perp$ と書こう．自然数の集合 $\mathbb{N}$ から $\mathbb{N} + \mathbf{1}$ への写像 $\lambda n.\ \perp_{\mathbb{N}}$ も $\perp$ で表わすことにします．こういうのも部分函数です

$$
\begin{aligned}
& G(\bot) \\
= \quad & \lambda n.\ (\textbf{if } n = 0 \textbf{ then } 1 \textbf{ else } n \times \bot(n \dot{-} 1) \\
= \quad & \lambda n.\ (\textbf{if } n = 0 \textbf{ then } 1 \textbf{ else } (1, *)
\end{aligned}
$$

図 6　$G(\bot)$ の計算

よね?

苦迦　もちろん．ちなみに今，羅茶さんが $\bot$ と記した部分函数は，先ほど紹介した部分函数の間の半順序 $\sqsubseteq$ に関する最小元です．つまり，集合 A から集合 B への部分函数の全体がなす集合 $[A, B]_p$ において，$\bot$ は $\sqsubseteq$ に関して最小の要素です．

羅茶　さて，$G(\bot)$ はどんな部分函数かな．図 6 のような計算をしてみました．つまり，$G(\bot)$ は仮引数 n が 0 の場合にだけ値が定義されていて，その値は 1，$n \neq 0$ の場合は値が定義されない，というような部分函数，ということになりました．

苦迦　結構です．ここで，$\bot \sqsubseteq G(\bot)$ であることに注意しておきましょう．

羅茶　あれ，どこかで見たような．そうだ，Tree の帰納的定義のときにも，$\varnothing \subseteq F(\varnothing)$ に注目しました．そういえば，$\bot$ も $\varnothing$ も最小元ですね．

苦迦　そうです．そうして，実は変換 G は F と同じく単調です．

羅茶　なるほど．それを確かめるのは後回しにして，感じをつかむために，$G(G(\bot))$ と，$G^3(\bot)$ くらいまでを計算してみましょう．図 7 のように計算できますね．どちらも，最後の段階で $n \times (\textbf{if } \ldots \textbf{ then } \ldots \textbf{ else } \ldots) = \textbf{if } \ldots \textbf{ then } n \times \ldots \textbf{ else } n \times \ldots$ と，if 文の中へ $n\times$ を繰り込みました．n が一つ増えるごとに定義域が一つずつ広くなりましたね．

苦迦　結構ですね．これで，$\bot \sqsubseteq G(\bot) \sqsubseteq G^2(\bot) \sqsubseteq G^3(\bot)$ が明らかになりました．

羅茶　そうですね．これで大体感じがつかめました．一般に $G^n(\bot) \sqsubseteq G^{n+1}(\bot)$ をいうためには，G が単調であること，つまり $f \sqsubseteq g$ であれば $G(f) \sqsubseteq G(g)$ であることを言えば十分ですね．

$$
\begin{aligned}
& G(G(\bot)) \\
= \quad & \lambda n.\ (\textbf{if } n = 0 \textbf{ then } 1 \textbf{ else } n \times G(\bot)(n \dot{-} 1)) \\
= \quad & \lambda n.\ (\textbf{if } n = 0 \textbf{ then } 1 \\
& \qquad \textbf{else } n \times \\
& \qquad\quad (\lambda m.\ (\textbf{if } m = 0 \textbf{ then } 1 \textbf{ else } \bot_{\mathbb{N}})(n \dot{-} 1)) \\
= \quad & \lambda n.\ (\textbf{if } n = 0 \textbf{ then } 1 \\
& \qquad \textbf{else } n \times (\textbf{if } n \dot{-} 1 = 0 \textbf{ then } 1 \textbf{ else } \bot_{\mathbb{N}})) \\
= \quad & \lambda n.\ (\textbf{if } n = 0 \textbf{ then } 1 \\
& \qquad \textbf{else } n \times (\textbf{if } n = 1 \textbf{ then } 1 \textbf{ else } \bot_{\mathbb{N}})) \\
= \quad & \lambda n.\ (\textbf{if } n = 0 \textbf{ then } 1 \\
& \qquad \textbf{else } mathbfif\ n = 1 \textbf{ then } n \times 1 \textbf{ else } n \times \bot_{\mathbb{N}}) \\
= \quad & \lambda n.\ (\textbf{if } n = 0 \textbf{ then } 1 \textbf{ else if } n = 1 \textbf{ then } 1 \textbf{ else } \bot_{\mathbb{N}}) \\
\\
& G^3(\bot) \\
= \quad & \lambda n.\ (\textbf{if } n = 0 \textbf{ then } 1 \textbf{ else } n \times G2(\bot)(n \dot{-} 1)) \\
= \quad & \lambda n.\ (\textbf{if } n = 0 \textbf{ then } 1 \\
& \qquad \textbf{else } n \times \\
& \qquad\quad (\lambda m.\ (\textbf{if } m = 0 \textbf{ then } 1 \\
& \qquad\qquad \textbf{else if } m = 1 \textbf{ then } 1 \textbf{else } \bot_{\mathbb{N}})(n \dot{-} 1)) \\
= \quad & \lambda n.\ (\textbf{if } n = 0 \textbf{ then } 1 \\
& \qquad \textbf{else } n \times (\textbf{if } n \dot{-} 1 = 0 \textbf{ then } 1 \\
& \qquad\qquad \textbf{else if } n \dot{-} 1 = 1 \textbf{ then } 1 \textbf{ else } \bot_{\mathbb{N}})) \\
= \quad & \lambda n.\ (\textbf{if } n = 0 \textbf{ then } 1 \\
& \qquad \textbf{else } n \times (\textbf{if } n = 1 \textbf{ then } 1 \\
& \qquad\qquad \textbf{else if } n = 2 \textbf{ then } 1 \textbf{ else } \bot_{\mathbb{N}})) \\
= \quad & \lambda n.\ (\textbf{if } n = 0 \textbf{ then } 1 \textbf{ else if } n = 1 \textbf{ then } n \times 1 \\
& \qquad \textbf{else if } n = 2 \textbf{ then } n \times 1 \textbf{ else } n \times \bot_{\mathbb{N}}) \\
= \quad & \lambda n.\ (\textbf{if } n = 0 \textbf{ then } 1 \textbf{ else if } n = 1 \textbf{ then } 1 \\
& \qquad \textbf{else if } n = 2 \textbf{ then } 2 \textbf{ else } \bot_{\mathbb{N}})
\end{aligned}
$$

図 7　$G(G(\bot))$，$G^3(\bot)$ の計算

この辺は先ほど F についてやったのと同じ筋書きですね．

苦迦　そのとおりで，その後も同じ筋書きが通用します．つまり，集合 $\{\, G^n(\bot) \mid n \in \mathbb{N} \,\}$ は上限を持ちます．

羅茶　先ほどは，$\{\, F^n(\varnothing) \mid n \in \mathbb{N} \,\}$ に対して，各要素 $F^n(\varnothing)$ の合併集合をとったのですが，今回は上限をとるのですね．上限とは何でしょう．

苦迦　上界のうち，最小のもののことです．$\{\, G^n(\bot) \mid n \in \mathbb{N} \,\}$ の上界とは，どの $G^n(\bot)$ よりも大きいものの事を言いますが，そういうものは一般には沢山あるかもしれません．沢山あるかもしれない上界のうちで最小のものを上限といいます．$\{\, F^n(\varnothing) \mid n \in \mathbb{N} \,\}$ の場合，実は合併集

合が $\subseteq$ に関する上限になってもいます.

羅茶 そうでしたか.

苦迦 どんな場合でも上限があるとは限りませんが,もしあるとすれば一つしかないので,集合 S の上限を $\mathrm{lub}\, S$ などと書きます.lub は Least Upper Bound の頭文字です.

羅茶 $\mathrm{lub}\, S$ と書いても,それが存在しなけりゃ幽霊みたいなものだ.

苦迦 今の場合,どの n についても $G^n(\bot) \sqsubseteq h$ が成り立ち,しかも,別の k が「どの n についても $G^n(\bot) \sqsubseteq k$」を満たせば,$\mathrm{lub}\{G^n(\bot) \mid n \in \mathbb{N}\} \sqsubseteq k$ であるような部分函数 $\mathrm{lub}\{G^n(\bot) \parallel n \in \mathbb{N}\}$ が存在する,といっているわけです.先ほど話題になったように,これは部分函数の昇列の上限ですから,全域的に定義できるとは限りませんが,この場合は運がよく,$\mathrm{lub}\{G^n(\bot) \mid n \in \mathbb{N}\}$ は全域函数になります.

羅茶 これがちょうど,我々が普通に言う階乗に一致するわけですね.

苦迦 はい.そうして,これも先ほどと同様に,$\mathrm{lub}\{G^n(\bot) \mid n \in \mathbb{N}\}$ は G の上限 (最小上界) であるばかりでなく,最小不動点であることがいえます.

羅茶 ああ,不動点であれば,自己参照的定義によって定義されるものと思うのが自然だと言うのは,先ほどの集合の帰納的定義の話と同じことなのですね.部分函数の変換であれば,どんな場合でも G のように最小不動点を持つのでしょうか.

苦迦 うーん,部分函数の変換を一般に考えると,とんでもない変換もありますよね.そういうものが最小不動点どころか,不動点を持つのかどうか,ちょっとわかりません.今回の議論では,G は単調なもの,と仮定しています.実はこの議論では単調であるだけでは不十分で,連続である必要があります.

羅茶 連続? 何でこんなところに連続函数がでてくるのかよくわかりませんが,ともあれ,どんな変換なら連続であって最小不動点をもつのか,ちょっと例をあげていただけませんか.

苦迦 まあ,函数の普通の定義の仕方なら大丈夫です.函数型プログラミング言語で函数を表現する式で表現される変換なら,最小不動点をもつと思ってよいでしょう.

羅茶 なるほど,集合の帰納的定義の場合は,函数型の構成が出てくると,ここでの議論が通用しませんでしたが,函数の再帰的定義の場合は,そういう大きな制限はなさそうですね.

3 Kleene の不動点定理

苦迦 ところで,集合の帰納的定義の議論と,函数の再帰的定義の議論では,共通する筋書きがありました.その核にあるのが,先ほどもちょっと名前を出した Kleene の定理です (図 8).

羅茶 うわ,これはまた,短いけれども知らない単語が入っていたり,知らない記法がはいっていたりして難物ですね.しかしとにかく,最小不動点があることを保証していることはわかります.順序関係を使ったのも,順序関係を得るために函数ではなく部分函数を持ち出すのも,すべてこの定理をうまく使うためだったのだ,というわけですね.

苦迦 はい.もともとは,自己参照的な定義を表わす変換の不動点こそが,その定義によって決められるものである,というのが始まりです.

羅茶 で,不動点といっても沢山あるかもしれないから,最小のものがあるのならそれにしておこうというわけですね.

苦迦 先ほども言ったように,最大のものが適当である場合もあるし,そもそも不動点が一つしかないような理論も立てることができるわけです.

羅茶 それにしても,知らない単語がいろいろで,くらくらします.連続写像というのは学生時代,解析で習いましたがね.

$A = (A, \sqsubseteq, \bot_A)$ を完備半順序集合とし,$F\colon A \to A$ を A から A への連続写像とする.このとき,$(F^j(\bot_A) \mid i \in \mathbb{N})$ は昇列であり,しかもその上限は F の最小不動点である.

図 8 Kleene の不動点定理

苦迦 ここで言う連続写像も，解析学でと同様に，位相空間の間の連続写像と言えなくもないのですが，位相空間の言葉を使わずに定義してしまえます．

羅茶 せっかく知ってる単語があったけれど，ますますわからなくなった……．ともあれ，Kleene の不動点定理を解読しよう．まず，完備半順序集合って何ですか？

苦迦 半順序関係はご存知でしたっけ？

羅茶 はい，反射的，反対称的で，推移的な二項関係ですね．

苦迦 そうです．関係 $\trianglelefteq$ が反射的というのは $x \trianglelefteq x$ がすべての x について成り立つこと，反対称的とは $x \trianglelefteq y$ しかも $y \trianglelefteq x$ ならば $x = y$ であること，推移的とは $x \trianglelefteq y$ しかも $y \trianglelefteq z$ であれば $x \trianglelefteq z$ であること，ですね．これが半順序関係です．集合と，そのうえの半順序関係をあわせて半順序集合といいます．「あわせて」という言葉をつかうより，集合と，その上の半順序関係の対を半順序集合という，というほうがいいでしょう．

羅茶 半順序関係は半順序集合の一部分，というか，構成要素の一つというわけですね．$\trianglelefteq$ が集合 A の上の半順序関係であるときに，A と $\trianglelefteq$ をあわせたもの $(A, \trianglelefteq)$ を半順序集合というわけだ．

苦迦 そうです．このときに A をこの半順序集合の台とか台集合と呼ぶ場合もあります．と言っておきながらなんですが，実際には半順序集合とその台を混同して用いますね．半順序集合の間の写像，などという言い方をしますがそれは実は両者の台集合の間の写像のことです．

羅茶 そういえば，半順序集合の間の順序を保つ写像に単調写像と言う名前がついていましたね．

苦迦 はい．$(A, \trianglelefteq)$ と $(B, \preccurlyeq)$ を半順序集合とするときに A から B への写像 $f\colon A \to B$ で $a \trianglelefteq a'$ ならば $f(a) \preccurlyeq f(a')$ が成り立つものを単調写像というのでした．

羅茶 ところで，例えば自然数の場合など，順序として大小関係を考えるのは当たり前なので，いちいち半順序関係 $\leq$ を指定するのは無駄な気がしますが．

苦迦 ああ，この辺も大切なことかもしれませんね．もし，その集合の上に，半順序関係が一つしかないのなら，半順序関係をいちいち指定する必要はないわけです．

羅茶 確かに．自然数の場合はどうですか．

苦迦 これはいい例でしたね．普通の大小関係 $\leq$ の他にも半順序関係を考えることができます．たとえば，「m は n を割り切る」あるいは「n は m の倍数である」と言う関係 $m \mid n$ は半順序関係です．

羅茶 ふむふむ．$m \mid m$ はいつも成り立つから反射的だし，$m \mid n$ と $n \mid m$ が両方成り立てば $m = n$ だな．だから反対称的．推移律も成り立ちますね．なるほど，確かにこれは $\leq$ とは全然違う半順序関係ですね．

苦迦 それどころか，実は何か半順序関係があれば，いつでも，それを使ってもう一つ半順序関係を導入することができます．双対といっていますが，$\leq$ から $\geq$ を導き出すようなものです．

羅茶 ああ，左右をひっくり返すだけですね．これも別のものですか？

苦迦 (m, n) と (n, m) は違いますから別のものですよね．こういうわけで，台集合を指定するだけでは，どの半順序関係を問題にしているのかが明らかだとは限らないのです．

羅茶 わかりました．

苦迦 しかし，羅茶さんのおっしゃることは，別の意味で尤もなことです．つまり，文脈によっては，どの半順序関係を考えているのかが，明らかである場合もあります．

羅茶 はい．そういう場合には，省略すると言うわけですか．

苦迦 我々の会話や文章の中では適宜省略していけばいいわけですね．しかし文章を計算機処理することを考えると，このような省略を厳密な形で定義することが大きな問題になります．

半順序集合などと数学のリテラシーの話をしているのかと思ったらいきなり自然言語処理のようなことになってきた．話があちこちに飛ぶのは大歓迎である．

羅茶 計算機に書かせるんなら，省略せずに書かせたらいいようにも思えますが．

苦迦 いえいえ，書くのは計算機がやってくれるかもしれませんが，それを人間が読むときに煩わしい，さらには，理解できなくなる，と言うことが問題です．また，人間が書いた文章を計算機にチェックさせる，と言うことも考えられます．その場合，人間は，分かりきったことは書きたくない．

羅茶 文章とおっしゃるから日本語や英語の文章かと思いましたが，確かに，形式言語での文章を人間が書く場合もあるわけだ．

苦迦 省略をどのようにするか，は実際に記述を書き下ろすことを，厳密に問題にするときに初めて出てくる問題の一つです．今の場合，記述の内容は数学ですが，それを書き下ろすことは必ずしも数学の問題ではありません．むしろそういうことこそ計算機科学固有の問題だと思うのですよ．

羅茶 ところで，なんでもかんでも集合と写像の言葉で表わすと言う方針で始めたのですが，早くも集合の上の「関係」なんていう，集合でも写像でもない言葉が出てきました．

大上段に振りかぶった苦迦の主張を，羅茶はあっさり無視して別の話題に移ってしまった．

苦迦 これはまあ，数学的リテラシーとでも言いたい科目の教科書に出てくる話題ではありますが，集合 A と B の間の関係を直積集合 $A \times B$ の部分集合のことである，とします．

羅茶 ああ，そうでした．a と b に関係がある，ということと (a, b) がその集合の要素になっている，ということを同じことだとみなすのでしたね．

苦迦 そうです．と説明したところで気づいたのですが，これはこれで一つの立場に過ぎませんね．計算機科学で大いに用いられている直観主義の立場では，a と b の間に関係がある，と言う結果だけではなく，その結果を得るにいたった証明までをも問題にします．

羅茶 ははあ．面白そうですね．ところで，半順序集合を思い出したところで，完備半順序集合に戻りましょう．

どんどん話が発散していく苦迦の話を，何なく軌道修正していく羅茶は，どこまでも醒めている．

苦迦 完備半順序集合とは，最小元をもつ半順序集合で，その中の昇列が必ず上限を持つもの，のことです．

羅茶 ふむふむ．例えば自然数全体の集合 $\mathbb{N}$ が普通の大小関係のもとでなす半順序集合 $(\mathbb{N}, \leq)$ は，完備半順序ではありませんね．最小元は 0 ですが，たとえば $f(i) \stackrel{\text{def}}{=} i$ によって定義される $\mathbb{N}$ の列 f は上限を持ちませんから．

苦迦 そのとおりです．しかし例えば，集合 S の部分集合の全体を S の冪集合といって，$\wp(S)$ と書きますが，$\wp(S)$ が集合の包含関係 $\subseteq$ に関してなす半順序集合 $(\wp(S), \subseteq)$ は，完備半順序です．空集合 $\varnothing$ が最小元ですし，どんな昇列 $X_1 \subseteq X_2 \subseteq X_3 \subseteq X_4 \subseteq \ldots$ を持ってきても，それらの合併集合 $\bigcup\{ X_i \,|\, i \in \mathbb{N}\}$ が，先ほどお話したように，$\{ X_i \,|\, i \in \mathbb{N} \}$ の上限ですから．

羅茶 そうでした．だから必ず上限が存在するわけですね．

苦迦 また，自然数の全体 $\mathbb{N}$ からそれ自身への部分函数の全体がなす集合 $[\mathbb{N}, \mathbb{N}]_p$ の上に，先ほど定義した関係 $\sqsubseteq$ を考えましょう．$([\mathbb{N}, \mathbb{N}]_p, \sqsubseteq)$ は半順序集合で，最小元 $\bot$ をもちます．

羅茶 $\bot$ はどの自然数に対しても定義されない部分函数でした．

苦迦 それに加えて，実は任意の昇列が上限を持ちます．

羅茶 あれ，そうですか．

苦迦 というわけで，$([\mathbb{N}, \mathbb{N}]_p, \sqsubseteq, \bot)$ はまたまた，完備半順序集合である，ということができます．実はこのことは完備半順序集合の冪の特別な場合とみなすことができます．

羅茶 ベキ？ ああ，そのような漢字を書くのですか．それにしてもまた難しい字ですね．常用漢字じゃあないんじゃありませんか，この冪という字．

苦迦 明治の数学者高木貞治でさえ，画数が多すぎると感じたらしく，巾という字を当てながら，「冪を巾と書くのは悪い癖である」と自分で書いています．いずれにしても，完備半順序集合の冪と言うのは，連続写像がつくる完備半順序集合のことです．先ほど冪集合 $\wp(S)$ が出てきました．今の冪

はこれと関係があるのですが，まずは別物と思ってください．詳しく述べてみましょう．A, B が完備半順序集合であるとき，A から B への連続写像の全体がつくる集合の上に，ある方法で完備半順序関係を定義することができます．その完備半順序集合を B^A あるいは $[A, B]$ などとかき，冪といいいます．

羅茶 なるほど．で，この場合，$([\mathbb{N}, \mathbb{N}]_p, \sqsubseteq, \bot)$ の台集合は，確かに写像の集合ではあるけれど，A や B にあたるモトの完備半順序集合が何なのかわからなくなりました．

苦迦 ちょっと説明が必要ですね．一般に集合 X にたいして，X の「平坦領域」という完備半順序集合 $X_\bot$ を考えることができます．

羅茶 あれ，さっきは $X + \mathbf{1}$ を $X_\bot$ と名づけていましたが．

苦迦 はい，実は今考える完備半順序集合の台集合がそれなのですよ．そうして，$\mathbb{N}_\bot$ から $\mathbb{N}_\bot$ への連続写像の全体が作る冪がちょうど，$([\mathbb{N}, \mathbb{N}]_p, \sqsubseteq, \bot)$ と同じものになっている，というわけなんです．

羅茶 なるほど．ところでお話に連続写像がでてきましたね．Kleene の定理にも出てきていました．完備半順序集合を問題にしているのに，なぜ連続写像がでてくるのですか．

苦迦 連続写像がちょうど，完備半順序集合の構造を保つ写像というべきものになっているからです．

羅茶 ふむふむ，ただの半順序集合の場合は，単調写像が，その構造を保つ写像でした．完備半順序集合は，半順序関係があることに加えて，最小元があることと，昇列の上限があることが要求されていますね．これらの「構造」を保つのが連続写像である，というわけなのですか．

苦迦 そうです．$f\colon A \to B$ が連続写像であるとは「$f(\bot_A) = \bot_B$，しかも任意の昇列 $a_1 \trianglelefteq a_2 \trianglelefteq a_3 \sqsubseteq \ldots$ に対して，$f(\mathrm{lub}\{a_i \mid i \in \mathbb{N}\}) = \mathrm{lub}\{f(a_i) \mid i \in \mathbb{N}\}$」ということです．

羅茶 なるほど．そういえば，実数の間の連続写像も，上限を保つものとして特徴づけられていましたね．名前の付け方には，納得できます．我々の連続写像は，半順序集合の間の写像なのですから，

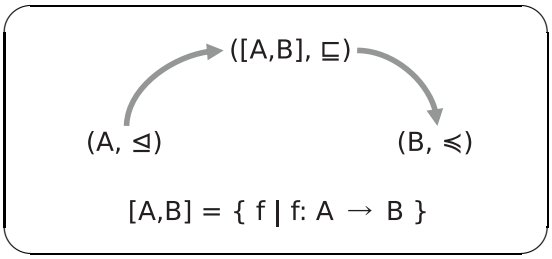

図 9　半順序集合の冪

ちょっと気になったのですが，連続写像は単調ですか？

苦迦 いい質問ですね．そのとおりです．実際，$a \trianglelefteq b$ であるとき，$x_1 = a$, $i \geq 2$ ならば $x_i = b$ として昇列 x_i を定義しましょう．f が連続であれば，$f(b) = f(\mathrm{lub}\{x_i \mid i \in \mathbb{N}\}) = \mathrm{lub}\{f(x_i) \mid i \in \mathbb{N}\} = \mathrm{lub}\{f(a), f(b)\}$ です．つまり $f(b)$ は $\{f(a), f(b)\}$ の上限です．したがって，$f(a) \preccurlyeq f(b)$ でなければなりません．

羅茶 なるほど．ところで，A から B への連続写像の全体がつくる集合の上に，決められた方法で完備半順序関係を定義することができる，ということでした．どういう風に決めるのでしょうか．

苦迦 まず連続函数のあいだに半順序関係を定義しなければなりませんね．実は，完備半順序集合の冪と同じようにして，半順序集合の冪 (図 9) というものを考えることができます．

羅茶 ははあ．半順序関係の構造を保つのは，単調写像だから，半順序集合の冪というと，単調写像の全体が作る半順序集合，というわけですか．ほんとにそんなものあるのかいな．

苦迦 幸か不幸か，この場合は，いつもちゃんとあります．まず，これを構成してみましょう．

羅茶 ふーむ，$(A, \trianglelefteq)$, $(B, \preccurlyeq)$ を半順序集合として，A から B への二つの単調写像 $f, g\colon A \to B$ の間にうまく順序をつけられるんだ，と言うわけか．どういう風に定義したらいいかな．何か A の要素 a をうまく持ってきて，f と g の関係を $f(a)$ と $g(a)$ の関係で決めてしまえばいい，なんて，これはちょっと乱暴かな．あ，そうか，一つだけの a じゃなくて，どんな a についても，とやれば

いいかな．A の任意の元 a について $f(a) \preccurlyeq g(a)$ のときに $f \sqsubseteq g$ と定義する．

苦迦 はい，この関係 $\sqsubseteq$ が単調写像全体の集合の上の半順序になっていることをいえばいいわけです．

羅茶 あ，そうか，それを証明しなくちゃいけないんだ．えーと，$f \sqsubseteq f$ だろうか．これはつまり，どんな a をもってきても $f(a) \preccurlyeq f(a)$ だろうか，と聞いているのと同じことだけれど，これは $\preccurlyeq$ が反射律を満たすから OK だ．じゃあ，$f \sqsubseteq g$ と $g \sqsubseteq f$ が成り立てば $f = g$ だろうか．a が与えられたとして，$f(a) \preccurlyeq g(a)$ であり，また，b も与えられたとして $g(b) \preccurlyeq f(b)$ であると仮定してみよう．この仮定のもとで，$f = g$ を示せばいいのだな．あれ，$f = g$ というのは，えーと，うーん，これはどう定義するべきでしょうね．

苦迦 ああ，そうですね．二つの写像が等しい，ということの定義はなんでしょうか．

羅茶 写像というのはつまるところ，定義域の要素を値域に写すものだから，同じものを同じものに写すときに二つの写像は等しい，とすればよいかな．今の場合だと定義域は A だから，A の任意の要素 a について $f(a) = g(a)$ のときに $f = g$ であると定義する．

苦迦 はい，ちょっと待ってください．実はここが一つの分かれ道なんですよ．

羅茶 天国と地獄？

苦迦 はは，それはわかりませんが，とにかく，「$f = g$ であれば，任意の $a \in A$ について $f(a) = g(a)$」というのは，文句のつけようのない，万人が認める命題です．羅茶さんの定義を採用すると，この逆，つまり「任意の $a \in A$ について $f(a) = g(a)$ であれば，$f = g$」も認めることになるわけです．これはもっともなことではありますが，こうすると，例えば値の計算の仕方が違っても同じ写像とみなす，ということになります．

羅茶 そうですね．写像の本質というのは何を何に写すかということであって，計算の仕方はどうでもいい，と思ったものですから．

苦迦 はい．それはそれで筋を通せばよいわけです．これは外延的立場ということができるでしょう．これに対して，写像には，「何を何に写すか」以外の情報も含まれる，とする立場もあり得ます．これは内包的立場です．例えば，配列の整列算法は，配列を配列に写す写像だとみなすことができますが，外延的立場では，算法つまり途中経過を問題にしませんから，いってみればバブルソートもクイックソートも等しくなります．算法が異なれば等しくないとみなすのは内包的立場です．

羅茶 なんだかややこしそうですね．とりあえずその，外延的立場，と言うやつで行きましょう．そっちのほうが簡単そうだ．そうすると，任意の c について $f(c) = g(c)$ が成り立つ，ということをいえばいい．c を一つとってくると，仮定から $f(c) \preccurlyeq g(c)$ も $g(c) \preccurlyeq f(c)$ も成り立つ．だから $\preccurlyeq$ についての反対称律によって $f(c) = g(c)$．なんなくできた．

苦迦 残るは推移律ですね．

羅茶 同じように証明できそうですよ．$f \sqsubseteq g$ と $g \sqsubseteq h$ を仮定して $f \sqsubseteq h$ をいえばいいのですね．そのためには任意の a について $f(a) \preccurlyeq h(a)$ をいえばいい．しかし，$f \sqsubseteq g$ から $f(a) \preccurlyeq g(a)$ がなりたち，$g \sqsubseteq h$ から $g(a) \preccurlyeq h(a)$ がなりたつ．したがって，$\preccurlyeq$ の推移律により $f(a) \preccurlyeq h(a)$ だが，これこそいいたかったことでした．

苦迦 お見事．こういうわけで半順序集合の間の単調写像の全体の上に半順序をうまく定義できることがわかりました．

羅茶 単にうまく定義できるだけでなく，写像に対して一般的に定義できる半順序はこれ以外にはなさそうですね．これ以外にない，と証明するのは難しそうですけれど．で，なんだか満たすべき性質のリストを全部証明して，一段落というところですが，何をしていたんだっけなあ．

苦迦 半順序集合 A から B への単調写像の全体の集合の上に関係を定義して，それが半順序関係であることを確かめたのですね．こうしてできる半順序集合 $[A, B]$ が，半順序集合の冪です．

羅茶 これで一段落．こんどは完備半順序集合の冪に話を進めましょう．完備半順序集合の間の連続写像の全体のうえに，うまく完備半順序関係を定義

しなければならないのですね.

苦迦 そうです．連続写像は単調ですから，先程と同じ定義によって，その間の半順序関係を定義することができます． $f \sqsubseteq g$ とは，すべての x について $f(x) \preccurlyeq g(x)$ と定めるわけです．そうして，この関係に関して最小の連続写像があることと，連続写像の昇列が上限をもつことを示すのです.

羅茶 $(A, \trianglelefteq, \bot_A)$ から $(B, \preccurlyeq, \bot_B)$ への連続写像で最小のものの候補として，何でもかんでも B の最小元 $\bot_B$ に写す写像を考えてみることにしてこれを $\bot$ と書くことにしよう．どの $a \in A$ についても $\bot(a) = \bot_B$ とするわけだ．$\bot$ が連続写像であることを確かめなくちゃ．$\bot$ が $\bot_A$ を $\bot_B$ に写すのは明らか．また，A の昇列は，$\bot_B$ のみからなる列に写されるわけだけれど，その上限が $\bot_B$ になるのは当然ですね．だから，$\bot$ は連続写像です．さてと，$\bot$ が半順序集合 $(A, \trianglelefteq)$ から $(B, \preccurlyeq)$ への単調写像のうちで最小のものであることは，その定義から明らかです．連続写像は単調写像ですから，$\bot$ が連続写像であれば，最小の連続写像でなくてはなりません.

苦迦 そうですね．これで，最小元があることが分かりました.

羅茶 今度は $[A, B]$ の昇列 $f_1 \sqsubseteq f_2 \sqsubseteq f_3 \sqsubseteq \ldots$ を考えて，これの上限を作ってみよう．あ，これは簡単．A の要素 a を，B の昇列 $f_1(a) \preccurlyeq f_2(a) \preccurlyeq f_3(a) \preccurlyeq \ldots$ の上限に写す写像を f としよう．f が $f_1 \sqsubseteq f_2 \sqsubseteq f_3 \sqsubseteq \ldots$ の上限になっていることはすぐわかりますね.

苦迦 これで，$(A, \trianglelefteq, \bot_A)$ から $(B, \preccurlyeq, \bot_B)$ への連続函数の集合がつくる完備半順序集合 $([A, B], \sqsubseteq, \bot)$ を構成することができました.

羅茶 これが完備半順序集合の冪というものですね．その上の連続写像か．あ，完備半順序集合の冪はまた完備半順序集合だから，そこから自分自身への連続写像，というものは確かに考えることができますね．ふう．なんで冪が問題になったかというと……あ，$\mathbb{N}_\bot$ から $\mathbb{N}_\bot$ 自身への連続写像の全体がつくる冪が，$\mathbb{N}$ から $\mathbb{N}$ への部分写像の全体がつくる完備半順序集合になっている，というこ

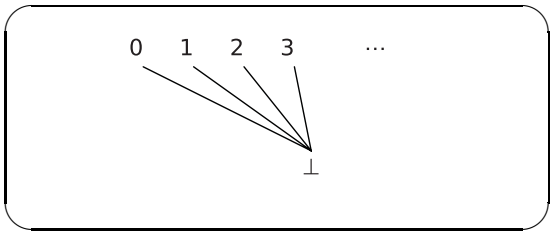

図 10　平坦領域 $\mathbb{N}_\bot$

となんだった．$\mathbb{N}_\bot$ の説明が，後のお楽しみということだったんでした.

苦迦 そうでした．さきほどは $\mathbb{N}_\bot$ で単なる集合 $\mathbb{N}+\mathbf{1}$ を表わしたのですが，これを台とする完備半順序集合 $(\mathbb{N}_\bot, \trianglelefteq, \bot_\mathbb{N})$ も同じく $\mathbb{N}_\bot$ と呼びます．ここで，$\bot_\mathbb{N} = (1, *)$ ですし，$\trianglelefteq$ は $x \trianglelefteq y \Leftrightarrow x = \bot_\mathbb{N}$ によって定義します．$*$ は $\mathbf{1}$ の唯一の要素でしたね.

羅茶 すると，$\bot_\mathbb{N}$ 以外の，いってみればホンモノの自然数同士は $\trianglelefteq$ の関係で結ばれないわけですね.

苦迦 そうです．図で書くと図 10 のようになります．こういうのを平坦領域 (flat domain) と呼んでいます．$\mathbb{N}_\bot$ は確かに完備半順序集合になっています.

羅茶 なるほど.

苦迦 で，$(\mathbb{N}_\bot, \trianglelefteq, (1, *))$ からこれ自身への連続写像ですが，これが丁度，$\mathbb{N}$ からそれ自身への部分写像に一致することを簡単に確かめることができます.

羅茶 昔，数学の先生に「簡単に証明できます」と言われたら気をつけろ，といわれたことがありました.

苦迦 ははは．学生が「……明らかに……」と言ったとたんに居眠りから覚めて，そこを追求する，と言う伝説をお持ちの先生もおられましたよ．話を進めましょう.

羅茶 ともあれ，そういう筋書きだったわけですね．そうすると，部分写像の周辺の定義は，冪のストーリに辻褄を合わせるように行われていたのでしょうか.

苦迦 そうです．むやみやたらに定義していたわけで

はありません．

羅茶 やれやれ，これでやっと Kleene の定理に現れている言葉の意味がわかりました．ついでに，集合や函数の自己参照的定義の説明に Kleene の定理がどういう風に使われるか，についても分かってきました．

苦迦 では，今度は Kleene の定理の証明の話をしましょう……．あっ，でも，もうすぐ来客があるんだった．

羅茶 今日はこの辺でおわりにしましょうかねえ．

苦迦 そうですね．でもせっかくだから，定理の証明を書いてしまいましょう．付録 A のようになります．

羅茶 これが自己参照的定義の，数学的な核というわけですね．

苦迦 そうです．

羅茶 また機会があったら，この証明について話を聞かせてください．函数の再帰的定義がピンと来ない，という私のボヤキから始まって，長い時間，函数の再帰的定義や集合の帰納的定義についての話になりました．

苦迦 函数の再帰的定義については，かなり一般的な議論ができたのですが，同じ議論では，集合の帰納的定義で函数型の構成を使ったものにまでは通用しませんでした．

羅茶 はい．集合の場合は $+$ と $\times$ だけからなる多項式のような変換だけということでしたね．

苦迦 函数型の話を進めるには，半順序集合を圏に一般化して議論することになります．

羅茶 あ，話の最後にまた耳慣れない言葉が出てきましたね．

苦迦 それについてはまたの機会に話すことにしましょう．そろそろ来客だ．私の部屋に戻ります．

羅茶 それではまた．

付録 A Kleene の不動点定理の証明

$(F^i(\bot)\,|\,i \in \mathbb{N})$ が昇列であることを示すために，$F^i(\bot) \leq F^{i+1}(\bot)$ を i に関する数学的帰納法で示す．

1. $i = 0$ のとき．$\bot$ は最小元だから，$F^0(\bot) = \bot \leq F^1(\bot)$.
2. $F^k(\bot) \leq F^{k+1}(\bot)$ を仮定して $F^{k+1}(\bot) \leq F^{k+2}(\bot)$ を示す．F は連続写像なので，単調である．このことと仮定から $F(F^k(\bot)) \leq F(F^{k+1}(\bot))$．この両辺を計算すると，$F^{k+1}(\bot) \leq F^{k+2}(\bot)$．これが示したいことであった．

次に，$(F^i(\bot)\,|\,i \in \mathbb{N})$ の上限 $\mathrm{lub}\{\,F^i(\bot)\,|\,i \in \mathbb{N}\,\}$ が F の不動点であること，つまり $F(\mathrm{lub}\{\,F^i(\bot)\,|\,i \in \mathbb{N}\,\}) = \mathrm{lub}\{\,F^i(\bot)\,|\,i \in \mathbb{N}\,\}$ を示す．まずこの左辺について考察する．F は連続であると仮定したので，$F(\mathrm{lub}\{\,F^i(\bot)\,|\,i \in \mathbb{N}\,\}) = \mathrm{lub}\{\,F(F^i(\bot)\,|\,i \in \mathbb{N})\,\} = \mathrm{lub}\{\,F^{i+1}(\bot)\,|\,i \in \mathbb{N}\,\}$ である．しかし，$\mathrm{lub}\{\,F^{i+1}(\bot)\,|\,i \in \mathbb{N}\,\} = \mathrm{lub}\{\,F^i(\bot)\,|\,i \in \mathbb{N}\,\}$ である．実際，$\{\,F^{i+1}(\bot)\,|\,i \in \mathbb{N}\,\}$ は $\{\,F^i(\bot)\,|\,i \in \mathbb{N}\,\}$ と $\{\,\bot\,\}$ の合併集合に過ぎない上，$\{\,F^i(\bot)\,|\,i \in \mathbb{N}\,\}$ の中に既に $\bot$ の上界が含まれている (つまり $\{\,F^i(\bot)\,|\,i \in \mathbb{N}\,\}$ は空ではない) から，両者の上限は一致する．したがって，$F(\mathrm{lub}\{\,F^i(\bot)\,|\,i \in \mathbb{N}\}) = \mathrm{lub}\{\,F^i(\bot)\,|\,i \in \mathbb{N}\,\}$，つまり $\mathrm{lub}\{\,F^i(\bot)\,|\,i \in \mathbb{N}\,\}$ は F の不動点である．

次に $\mathrm{lub}\{\,F^i(\bot)\,|\,i \in \mathbb{N}\,\}$ は F の不動点のうち最小であることを示す．そのために $F(f) = f$ つまり f が F の不動点であると仮定して，$\mathrm{lub}\{\,F^i(\bot)\,|\,i \in \mathbb{N}\,\} \leq f$ を示そう．f が $\{\,F^i(\bot)\,|\,i \in \mathbb{N}\,\}$ の上界であることをいえばよい．任意の自然数 j について $F^j(\bot) \leq f$ であることを，j に関する数学的帰納法で示す．$j = 0$ のときは $F^0(\bot) = \bot \leq f$．次に $F^k(\bot) \leq f$ を仮定する．このことと，F が連続，したがって単調であることから，$F(F^k(\bot)) \leq F(f)$，しかし f は F の不動点なので $F(f) = f$．したがって，$F^{k+1}(\bot) \leq f$.

以上で，$\mathrm{lub}\{\,F^i(\bot)\,|\,i \in \mathbb{N}\,\}$ が F の不動点のうちで最小であることが示された．

木下佳樹

東京大学理学部情報科学科卒業．テキサスインスツルメンツ勤務の後，同大学大学院理学系研究科情報科学専攻博士課程修了．理学博士．電子技術総合研究所情報アーキテクチャ部言語システム研究室研究員，

産業技術総合研究所システム検証研究センター長等を経て現在同情報技術研究部門主幹研究員兼奈良先端科学技術大学院大学情報科学研究科連携教授．本会では企画委員，理事，ディペンダブルシステム研究会主査を務めた．