再帰的関数論

照井一成
京都大学数理解析研究所
terui@kurims.kyoto-u.ac.jp
http://www.kurims.kyoto-u.ac.jp/~terui

1　はじめに

再帰的関数論（recursion theory）とは？
計算理論の一種。帰納的関数論とも訳される。より包括的な観点から計算についての研究を行う**計算可能性理論**（computability theory）としばしば同一視される。計算可能性理論とは、ごく大雑把に言って（数についての）性質や関係、関数などを計算という観点から分析する理論である。これが再帰的関数論と呼ばれるときには、**再帰法**（recursion）による関数定義という側面が念頭に置かれていることが多い。

スコーレムらの先行研究の後、ゲーデルによる第一不完全性定理を証明において主要な道具立ての一つとして用いられた。その後エルブラン、チャーチ、ロッサー、クリーニ、テューリングらの研究により１９３０年代に大きく発展した。２０世紀後半に入り、ハードウェアとしてのコンピュータが現実味を帯びてくると、それまでのような「理念としての計算」という観点のみならず、「実用としての計算」という観点も重要になる。オートマトン理論、形式言語理論、計算の複雑さの理論（しいては計算機科学全般）などは全て計算可能性理論から派生したものと見なすことができる。

計算可能性理論で取り扱われる典型的な問い

- 計算するとはどういうことか？
- 関数が計算可能であるとはどういうことか？
- どのような関数が計算（不）可能か？
- 計算（不）可能な関数はどのように分類できるか？
- 計算（不）可能な関数はどのような性質を持つか？

ここでは主に自然数を用いた計算、自然数についての関係や関数に限定して話を進める。

2　簡単な集合論の準備

集合論においてもっとも基本的なのは「対象 a が集合 A の要素である」という関係であり、このことを

$$a \in A$$

というように記す。「a は A の元である」、「a は A に属する」というような言い方もする。これの否定「対象 a は集合 A の要素ではない」は

$$a \notin A$$

と記す。本稿では対象ということで主に**自然数** $0, 1, 2, \ldots$ が想定されている。自然数全体の集合を $\mathbb{N}$ と書くことにする。すなわち、$0 \in \mathbb{N}$, $1 \in \mathbb{N}$, $2 \in \mathbb{N}, \ldots$ である。

対象 $a_1, \ldots, a_k$ からなる集合を

$$\{a_1, \ldots, a_k\}$$

と記す。すると、もちろん、$a_i \in \{a_1, \ldots, a_k\}$ が各 $1 \leq i \leq k$ について成り立つ。

$\varphi(x)$ を自然数に関する性質とするとき、$\varphi(x)$ を満たす自然数全ての集合（$\varphi(x)$ の**外延**）を

$$\{x|\varphi(x)\}$$

と書いて表す（より正確には $\{x \in \mathbb{N} \mid \varphi(x)\}$ と書くべき）。すると任意の $n \in \mathbb{N}$ について、

$$\varphi(n) \Longleftrightarrow n \in \{x|\varphi(x)\}$$

が成り立つ。

集合論の重要な原理の一つに**外延性の原理** (extensionality principle) がある。これは、「要素がすべて等しいような 2 つの集合は等しい」「集合の同一性はその要素のみにより定められる」ことを述べるものであり、より形式的には、

$$A = B \Longleftrightarrow \text{for all } x(x \in A \text{ iff } x \in B)$$

と表される。

空集合 (the empty set) とは、元を一つも含まない集合のことであり、ϕ と記される。空集合は、

$$\text{For all } x(x \notin \phi)$$

という性質を持つ。この性質を持つ集合はみな同一となることが外延性の原理より導かれる。つまり、空集合はただ一つしか存在しない。

「集合 A は集合 B の**部分集合** (subset) である」ことを $A \subseteq B$ と書く。正確な定義は、

$$A \subseteq B \Longleftrightarrow \text{ for all } x(x \in A \text{ implies } x \in B)$$

である。特に任意の集合 A について

$$\phi \subseteq A$$

であることは容易に確かめられる。また、外延性の原理より、

$$A \subseteq B \text{ and } B \subseteq A \Longleftrightarrow A = B$$

が成り立つ。

集合 A と B の**交わり** (intersection)、**和** (union)、**差** (difference) はそれぞれ $A \cap B$、$A \cup B$、$A - B$ と表され、次のように定義される。

$$a \in A \cap B \Longleftrightarrow a \in A \text{ and } a \in B$$

$$a \in A \cup B \Longleftrightarrow a \in A \text{ or } a \in B$$

$$a \in A - B \Longleftrightarrow a \in A \text{ and } a \notin B.$$

すると次の性質が成り立つ。

1. $A \cap B \subseteq A$.
2. $A \subseteq A \cup B$.
3. $C \subseteq A$ and $C \subseteq B \Longrightarrow C \subseteq A \cap B$.
4. $A \subseteq C$ and $B \subseteq C \Longrightarrow A \cup B \subseteq C$.

対象 a と b の**順序対** (ordered pair) を $\langle a, b \rangle$ と記す。順序対について重要なのは、

$$\langle a, b \rangle = \langle a', b' \rangle \Longleftrightarrow a = a' \text{ and } b = b'$$

という性質である。$\{a, b\}$ と $\langle a, b \rangle$ の違いに注意。外延性の原理より $\{a, b\} = \{b, a\}$ であるが、一般に $\langle a, b \rangle = \langle b, a \rangle$ であるとは限らない。

順序対の概念は、3 つ組 $\langle a_1, a_2, a_3 \rangle$、4 つ組 $\langle a_1, a_2, a_3, a_4 \rangle$ 等に自然に拡張できる。特に 1 つ組 $\langle a_1 \rangle$ とは a_1 そのもののことであるとする。

自然数の k 個組全ての集合のことを $\mathbb{N}^k$ と書いて表す。すなわち、

$$\mathbb{N}^k = \{\langle n_1, \ldots, n_k \rangle \mid n_1 \in \mathbb{N}, \ldots, n_k \in \mathbb{N}\}$$

上の取り決めにより、$\mathbb{N}^1 = \mathbb{N}$ である。

自然数上の k **項関係** R とは、$\mathbb{N}^k$ の部分集合のことである。特に 1 項関係 $R \subseteq \mathbb{N}$ とは $\mathbb{N}$ の部分集合のことであり、自然数についての**性質**とも呼ばれる。$\langle n_1, \ldots, n_k \rangle \in R$ が成り立つとき、$R(n_1, \ldots, n_k)$ と書く。また、R が 2 項関係の時には、$\langle n_1, n_2 \rangle \in R$ のことを $n_1\ R\ n_2$ とも書く (infix notation)。

例えば、大小関係 $<$ は集合論的には $\{\langle x, y \rangle |$ y は x よりも大きい $\}$ で表現できる。これは $\mathbb{N}^2$ の部分集合であり、$\langle 1, 2 \rangle$、$\langle 3, 7 \rangle$ 等を要素として含む。一般に $\langle 1, 2 \rangle \in <$ などと書く代わりに infix notation を用いて $1 < 2$ と書く。

自然数上の k **項関数** f とは、$\mathbb{N}^{k+1}$ の部分集合で次の性質を満たすもののことである：

> (*) 任意の $\langle n_1, \ldots, n_k \rangle \in \mathbb{N}^k$ について $\langle n_1, \ldots, n_k, m \rangle \in f$ となるような $m \in \mathbb{N}$ がただ一つ存在する。

このただ一つ存在する m のことを $\langle n_1, \ldots, n_k \rangle$ に対する f の**値** (value) と呼び、$f(n_1, \ldots, n_k)$ と書く。f が（自然数上の）k 項関数であることを、

$$f : \mathbb{N}^k \longrightarrow \mathbb{N}$$

と書いて表す。

例えば、足し算 $+$ は集合論的には $\{\langle x,y,z\rangle |\ x+y=z\}$ で表現できる。これは確かに関数の定義に合致しており、実際、どのような $\langle n_1,n_2\rangle$ が与えられても、$\langle n_1,n_2,m\rangle \in +$ を満たす m が唯一つ存在する。この値 m とは n_1+n_2 のことに他ならない。

上の定義の特別な場合として、**0 項関数**とは自然数のことであると取り決めておく。すなわち、$0,1,2\ldots$ は自然数であると同時に 0 項関数でもある。

自然数上の k **項部分関数** f とは、$\mathbb{N}^{k+1}$ の部分集合で次の性質を満たすもののことである:

> (*) $\langle n_1,\ldots,n_k,m\rangle \in f$ となるような $m\in\mathbb{N}$ が存在するときには、そのような m は唯一つに限る。

$\langle n_1,\ldots,n_k\rangle$ に対してそのような m が存在するとき、関数 f は $\langle n_1,\ldots,n_k\rangle$ において**定義されている**という。そのような m を $\langle n_1,\ldots,n_k\rangle$ に対する f の**値** (value) と呼び、$f(n_1,\ldots,n_k)$ と書いて表す。

関数と部分関数の違いは、関数が全ての $\langle n_1,\ldots,n_k\rangle \in \mathbb{N}^k$ に対して定義されているのに対して、部分関数はそうとは限らないということである（部分関数と区別するために関数はしばしば**全域関数**と呼ばれる）。一方で、ある $\langle n_1,\ldots,n_k\rangle$ に対して部分関数が定義されているときには、その値は唯一つであるという点で関数と共通している。

例えば、引き算 $-$ は集合論的には $\{\langle x,y,z\rangle |\ x-y=z\}$ で表現できる。$n\geq m$ を満たす $\langle n,m\rangle \in \mathbb{N}^2$ に対して引き算は定義されており、その値は $n-m$ である。そうでないような $\langle n,m\rangle \in \mathbb{N}^2$ に対しては引き算は定義されていない。同様に、自然数上の割り算も 2 項部分関数と見なすことができる。

自然数上の集合 $R\subseteq\mathbb{N}$ が与えられたとき、その**特性関数** (characteristic function) χ_R : $\mathbb{N}\longrightarrow\mathbb{N}$ とは次のように定義される関数のことである：

$$\begin{aligned}\chi_R(n) &= 1\ (R(n)\text{ が成り立つとき})\\ &= 0\ (R(n)\text{ が成り立たないとき})\end{aligned}$$

例えば、集合 $Even=\{x|\ x\text{ は偶数である }\}$ の特性関数 χ_{Even} とは次のような関数のことである：

$$\begin{aligned}\chi_{Even}(n) &= 1\ (n\text{ が偶数のとき})\\ &= 0\ (n\text{ が奇数のとき})\end{aligned}$$

より一般的に、自然数上の k 項関係 $R\subseteq\mathbb{N}^k$ が与えられたとき、その特性関数 $\chi_R:\mathbb{N}^k\longrightarrow\mathbb{N}$ は次のように定義される：

$$\begin{aligned}\chi_R(n_1,\ldots,n_k) &= 1\ (R(n_1,\ldots,n_k)\text{ が成り立つとき})\\ &= 0\ (R(n_1,\ldots,n_k)\text{ が成り立たないとき})\end{aligned}$$

逆に、値が常に 0 か 1 であるような k 項関数 f （そのような関数を $f:\mathbb{N}^k\longrightarrow\{0,1\}$ と書いて表す）が与えられたとき、集合

$$\hat{f}=\{\langle x_1,\ldots,x_k\rangle | f(x_1,\ldots,x_k)=1\}$$

を考えることができる。$\hat{f}$ は自然数上の k 項関係であり、任意の $R \subseteq \mathbb{N}^k$、任意の $f : \mathbb{N}^k \longrightarrow \{0,1\}$ について次の性質が成り立つ：

$$\hat{\chi}_R = R, \quad \chi_{\hat{f}} = f$$

このことから、自然数上の k 項関係の集合と、$\{0,1\}$ を値域とする自然数上の関数（$f : \mathbb{N}^k \longrightarrow \{0,1\}$ となる関数）の集合は一対一に対応することがわかる。

練習問題 2.1 $\hat{\chi}_R = R$, $\chi_{\hat{f}} = f$ が成り立つことを確かめよ。

3 原始再帰的関数

再帰的関数を研究する手始めとして、本章ではまず、より基本的な原始再帰的関数について考える。また原始再帰的関係についても言及する。原始再帰的関数とは、基本的な関数から始めて関数合成や原始再帰法（primitive recursion）を繰り返し適用することにより得られる関数のことである。そのようにして得られる関数が計算可能であることは比較的容易に理解できる。基本的関数はみな計算可能であることが明らかなものばかりであり、関数合成や原始再帰法は計算可能性を保存するからである。

原始再帰法による関数構成が計算可能性を保つという点について簡単に補足しておく。原始再帰法とは数論における漸化式（recursive formula）による数列の定義を一般化したものである。漸化式による数列 $a_0, a_1, a_2, \ldots$ の定義とは例えば

$$\begin{aligned} a_0 &= 5 \\ a_{n+1} &= 2a_n + 3 \end{aligned}$$

のようなもので、このとき $a_0 = 5$ であることは定義通りであるし、$a_1 = 13$、$a_2 = 29$、$a_3 = 61$ であることは第二式を繰り返し適用すればすぐにわかる。一般に、どんな a_n の値も第二式を必要な回数だけ繰り返すことにより求めることができる。原始再帰法とはこのような数列（関数）の定義法を一般化したものに他ならず、そのようにして定義される関数が計算可能であることは同様の議論により理解することができる。

いかに基本的であるとはいえ、原始再帰的関数のクラスは広大である。実際、数論において現れる関数（関係）やプログラマが取り扱う関数（関係）の大部分は原始再帰的である。にもかかわらず、後ほど例を挙げるように、全ての計算可能な関数が原始再帰的であるわけではない。直感的に言って、原始再帰的関数については「インプットが与えられたならば、そのインプットの大きさから計算にどれくらいの時間がかかるか（何兆世紀かかるか）」が比較的容易に見積り可能である。先ほどの漸化式の場合のように、何回第二式（ステップ関数）を繰り返せばよいかが事前に見積り可能だからである。一方、世の中には計算可能であっても計算にどれくらいの時間がかかるか（いつ計算が終わるか）が全く予想できない関数というものも存在する。そのような関数はたとえ計算可能であっても原始再帰的ではありえない。

3.1 原始再帰的関数

定義 3.1 原始再帰的関数 *(primitive recursive functions)* を次のように定義する。

1. **ゼロ関数** *(zero function)* : 各 $n \in \mathbb{N}$ について n 項関数 $\mathsf{zero}^n(x_1, \ldots, x_n) = 0$ は原始再帰的である。
2. **射影関数** *(projection functions)* : $1 \leq i \leq n$ を満たす各 $n, i \in \mathbb{N}$ について n 項関数 $\mathsf{proj}_i^n(x_1, \ldots, x_n) = x_i$ は原始再帰的である。
3. **後続者関数** *(successor function)* : 一項関数 $\mathsf{suc}(x) = x + 1$ は原始再帰的である。

ゼロ関数、射影関数、後続者関数を合わせて**初期関数**と呼ぶ。

4. **合成** *(composition)* : g が m 項原始再帰的関数であり、$h_1, \ldots, h_m$ が n 項原始再帰的関数ならば、

$$f(x_1, \ldots, x_n) = g(h_1(x_1, \ldots, x_n), \ldots, h_m(x_1, \ldots, x_n))$$

により定義される関数 f は原始再帰的である。ここで $1 \leq m, 0 \leq n$。

5. **原始再帰法** *(primitive recursion)* : g が n 項原始再帰的関数であり、h が $n+2$ 項原始再帰的関数ならば、

$$\begin{aligned} f(x_1, \ldots, x_n, 0) &= g(x_1, \ldots, x_n) \\ f(x_1, \ldots, x_n, y+1) &= h(x_1, \ldots, x_n, y, f(x_1, \ldots, x_n, y)) \end{aligned}$$

により定義される $n+1$ 項関数 f は原始再帰的である。ここで $0 \leq n$。g は**基底関数** *(base function)* と呼ばれ、h は**（再帰）ステップ関数** *(recursion step function)* と呼ばれる。また、h の第 $n+1$ 項（y が代入されている項）は**後退項** *(regressive argument)* と呼ばれ第 $n+2$ 項（$f(x_1, \ldots, x_n, y)$ が代入されている項）は**再帰項** *(recursive argument)* と呼ばれる。

以上の定義により、足し算 $x + y$ は原始再帰的関数となる。このことは次のように確かめることができる。

- proj_1^1 は 2. により（1 項）原始再帰的関数である。
- proj_3^3 は 2. により（3 項）原始再帰的関数である。
- suc は 3. により（1 項）原始再帰的関数である。
- $h_0(x_1, x_2, x_3) = \mathsf{suc}(\mathsf{proj}_3^3(x_1, x_2, x_3))$ により定義される 3 項関数 h_0 は 4. により原始再帰的である。
- 次のように定義される 2 項関数 plus は 5. により原始再帰的である：

$$\begin{aligned} \mathsf{plus}(x, 0) &= \mathsf{proj}_1^1(x) \\ \mathsf{plus}(x, y+1) &= h_0(x, y, \mathsf{plus}(x, y)) \end{aligned}$$

このようにして得られる plus は確かに足し算を表す。例えば、$\mathsf{plus}(3,2)$ を定義に従って等式変形すると、次のようになる：

$$\begin{aligned}
\mathsf{plus}(3,2) &= h_0(3,1,\mathsf{plus}(3,1)) \\
&= \mathsf{suc}(\mathsf{proj}^3_3(3,1,\mathsf{plus}(3,1))) \\
&= \mathsf{suc}(\mathsf{plus}(3,1)) \\
&= \mathsf{suc}(h_0(3,0,\mathsf{plus}(3,0))) \\
&= \mathsf{suc}(\mathsf{suc}(\mathsf{proj}^3_3(3,0,\mathsf{plus}(3,0)))) \\
&= \mathsf{suc}(\mathsf{suc}(\mathsf{plus}(3,0))) \\
&= \mathsf{suc}(\mathsf{suc}(\mathsf{proj}^1_1(3))) \\
&= \mathsf{suc}(\mathsf{suc}(3)) \\
&= \mathsf{suc}(4) \\
&= 5
\end{aligned}$$

ゆえに $\mathsf{plus}(3,2) = 5$ であることが確かめられた。plus が足し算を表すことをより正確に検証するためには、数学的帰納法による証明を行う必要がある（次の練習問題の 2. を参照）。

練習問題 3.2

1. $\mathsf{plus}(4,3) = 7$ となることを確かめよ。
2. *(*)* 任意の自然数 n, m について、$\mathsf{plus}(n,m) = n + m$ となることを証明せよ。［ヒント：m に関する数学的帰納法による。］

同様にして、掛け算 $x \cdot y$ が原始再帰的関数であることは次のようにして確かめることができる。

- zero^1 は 1. により（1 項）原始再帰的関数である。
- plus は先ほど示したとおり（2 項）原始再帰的関数である。
- proj^3_1, proj^3_3 は 2. により（3 項）原始再帰的関数である。
- $h_1(x_1,x_2,x_3) = \mathsf{plus}(\mathsf{proj}^3_1(x_1,x_2,x_3), \mathsf{proj}^3_3(x_1,x_2,x_3))$ により定義される 3 項関数 h_1 は 4. により原始再帰的である。
- 次のように定義される 2 項関数 times は 5. により原始再帰的である：

$$\begin{aligned}
\mathsf{times}(x,0) &= \mathsf{zero}^1(x) \\
\mathsf{times}(x,y+1) &= h_1(x,y,\mathsf{times}(x,y))
\end{aligned}$$

練習問題 3.3

1. $\mathsf{times}(4,3) = 12$ となることを確かめよ。

2. *(*)* 任意の自然数 n, m について、$\mathsf{times}(n, m) = n \cdot m$ となることを証明せよ。［ヒント：m に関する数学的帰納法による。］

3. べき乗関数 x^y が原始再帰的であることを確かめよ。

4. 階乗関数 $x!$ が原始再帰的であることを確かめよ。

定義 3.1 における原始再帰的関数の定義はコンパクトではあるが、いろいろな関数の原始再帰性を示すときにはやや使いづらい。そこでそのような際に便利な補題をいくつか証明しておく。

補題 3.4 f が原始再帰的関数ならば、次のように定義される関数 f' も原始再帰的である：

$$f'(x_1, \ldots, x_n) = f(x_{i_1}, \ldots, x_{i_m})$$

ここで $i_1, \ldots, i_m$ はそれぞれ $\{1, \ldots, n\}$ の要素である。

最後の条件が述べているのは、右辺に現れる変数は全て左辺にも現れるということである。この条件が成り立つような等式により定義される限り、f' は常に原始再帰的となるというのが上の補題の主張である。

例えば次のように定義される平方関数 square は原始再帰的であることがわかる：

$$\mathsf{square}(x) = \mathsf{times}(x, x)$$

また、y の値に関係なく常に $2x$ を返す関数 twice' も原始再帰的であることがわかる：

$$\mathsf{twice}'(x, y) = \mathsf{plus}(x, x)$$

証明 f' は f と $\mathsf{proj}^n_{i_1}, \ldots, \mathsf{proj}^n_{i_m}$ から合成により定義することができる：

$$f'(x_1, \ldots, x_n) = f(\mathsf{proj}^n_{i_1}(\vec{x}), \ldots, \mathsf{proj}^n_{i_m}(\vec{x}))$$

ここで $\vec{x} = x_1, \ldots, x_n$。ゆえに f' は原始再帰的である。 ■

上の補題の帰結として、次のような操作は原始再帰的関数を構成する際に自由に行ってもよいことになる：$f(x_1, \ldots, x_n)$ が原始再帰的関数であるとするならば、次のように定義される f_1, f_2, f_3 も原始再帰的関数である。

1. 余剰項の追加（weakening）：$f_1(x_1, \ldots, x_n, y) = f(x_1, \ldots, x_n)$

2. 同一項の複数回使用（contraction）：$f_2(x_1, \ldots, x_{n-2}, y) = f(x_1, \ldots, x_{n-2}, y, y)$

3. 項の順番の入れ替え（exchange）：$f_3(x_1, \ldots, x_{n-2}, y, z) = f(x_1, \ldots, x_{n-2}, z, y)$

上の補題のさらなる帰結として、合成や原始再帰法は次のように制限の緩和された、より使いやすい形と置き換えることができることになる：

補題 3.5

1. **自由合成**：$\vec{y}, \vec{z}, \vec{w}$ を変数の列とする。g, h が原始再帰的ならば、次のように定義される関数 f も原始再帰的である：

$$f(x_1, \ldots, x_n) = g(\vec{y}, h(\vec{z}), \vec{w})$$

ここで右辺に現れる変数 $\vec{y}, \vec{z}, \vec{w}$ はみな左辺にも現れるものとする（すなわち $x_1, \ldots, x_n$ のどれかと等しい）。

2. **自由原始再帰法**：g, h が原始再帰的関数ならば、次のように定義される関数 f も原始再帰的である：

$$\begin{aligned} f(x_1, \ldots, x_n, 0) &= g(\vec{z}) \\ f(x_1, \ldots, x_n, y+1) &= h(\vec{w}, y, f(x_1, \ldots, x_n, y)) \end{aligned}$$

ここで右辺に現れる変数 $\vec{z}, \vec{w}$ はそれぞれ $x_1, \ldots, x_n$ のどれかである。さらに、下式右辺において、再帰項 $f(y, x_1, \ldots, x_n)$ と後退項 y は実際には何度現れていてもよく、どの順でどこに現れていてもよく、どちらか一方が（または両方とも）現れていなくともよい。

証明　1. 上の補題より、次のように定義される g', h' は原始再帰的である：

$$\begin{aligned} g'(x_1, \ldots, x_n, v) &= g(\vec{y}, v, \vec{w}) \\ h'(x_1, \ldots, x_n) &= h(\vec{z}) \end{aligned}$$

f は g' と h' と $\mathsf{proj}^n_1, \ldots, \mathsf{proj}^n_n$ から合成により定義することができる：

$$f(x_1, \ldots, x_n) = g'(\mathsf{proj}^n_1(\vec{x}), \ldots, \mathsf{proj}^n_n(\vec{x}), h'(\vec{x}))$$

ここで $\vec{x} \equiv x_1, \ldots, x_n$ である。ゆえに f は原始再帰的である。

2. 練習問題とする。 ■

すなわち、合成により関数 $f(x_1, \ldots, x_n)$ を定義する場合、$x_1, \ldots, x_n$ の中には右辺では使われない変数があってもよいし、何度も使われる変数があってもよい。また、変数が使われる順番は関係ない。原始再帰法により関数 $f(y, x_1, \ldots, x_n)$ を定義する場合にも $x_1, \ldots, x_n$ について同様のことが成り立つ。加えて、原始再帰法による定義の第二式においては、再帰項や後退項は用いても用いなくともよい。

自由原始再帰法を使えば、足し算、掛け算が原始再帰的であることは直接的に示すことができる：

$$\begin{aligned} \mathsf{plus}(x, 0) &= x \\ \mathsf{plus}(x, y+1) &= \mathsf{suc}(\mathsf{plus}(x, y)) \\ \mathsf{times}(x, 0) &= 0 \\ \mathsf{times}(x, y+1) &= \mathsf{plus}(x, \mathsf{times}(x, y)) \end{aligned}$$

第一式右辺における x は $\mathsf{proj}^1_1(x)$ の略記であり、第三式右辺における 0 は zero^0 の略記である（zero^0 は 0 項の関数と見なされている点に注意）。以下、同様の略記を断りなく用い

る。第二式、第四式においては後退項 y は使われていない。このような定義によっても原始再帰的関数が得られることは、上の補題により保証されている。

今後は通常の表記法に従って、$\mathsf{plus}(x,y)$, $\mathsf{times}(x,y)$ を $x+y$, $x\cdot y$ と書く。以後同様の infix notation を断りなく用いることにする。また、$\vec{x}, \vec{y}$ などのベクトル表記は常に変数の列を表すものとする。

命題 3.6 以下に挙げる関数は全て原始再帰的である。

1. 定数関数 *(constant functions)*：各 $p, n \in \mathbb{N}$ について、$\mathsf{const}^n_p(x_1, \ldots, x_n) = p$.
2. 前者関数 *(predecessor function)*：

$$\begin{aligned} \mathsf{pred}(0) &= 0 \\ \mathsf{pred}(y+1) &= y \end{aligned}$$

3. 切り捨てありの引き算 *(truncated subtraction)*：

$$\begin{aligned} x \dot{-} y &= x - y \quad (x \geq y \text{ のとき}) \\ &= 0 \quad (x < y \text{ のとき}) \end{aligned}$$

4. 正数テスト：

$$\begin{aligned} \mathsf{pos?}(0) &= 0 \\ \mathsf{pos?}(y+1) &= 1 \end{aligned}$$

5. ゼロテスト：

$$\begin{aligned} \mathsf{zero?}(0) &= 1 \\ \mathsf{zero?}(y+1) &= 0 \end{aligned}$$

6. 差：$|x-y|$

証明 1. $\mathsf{const}^n_p(\vec{x}) = \underbrace{\mathsf{suc}(\cdots\mathsf{suc}}_{p\ \ times}(\mathsf{zero}(\vec{x}))\cdots)$。
2. 自由原始再帰法により定義されているので明らか。第二式右辺においては、再帰項は用いられず、後退項のみが用いられている点に注意。より正確に書けば、$\mathsf{pred}(y+1) = \mathsf{proj}^2_1(y, \mathsf{pred}(y))$ となる。
3. 自由原始再帰法により次のように定義できる：

$$\begin{aligned} x \dot{-} 0 &= x \\ x \dot{-} (y+1) &= \mathsf{pred}(x \dot{-} y) \end{aligned}$$

4. $\mathsf{pos?}(y) = y \dot{-} \mathsf{pred}(y)$。
5. $\mathsf{zero?}(y) = 1 \dot{-} \mathsf{pos?}(y)$。
6. $|x-y| = (x \dot{-} y) + (y \dot{-} x)$。 ■

練習問題 3.7 *(*)* 最大値関数 $\mathsf{max}(x,y)$（x と y のうち大きい方を返す関数）、最小値関数 $\mathsf{min}(x,y)$（x と y のうち小さい方を返す関数）が原始再帰的関数であることを示せ。［ヒント：$+$ と $\dot{-}$ を合成する。］

命題 3.8 $n+1$ 項原始再帰的関数 $f(\vec{x},y)$（ここで $\vec{x} \equiv x_1,\ldots,x_n$）が与えられたとき、次のように定義される関数も $n+1$ 項原始再帰的関数である：

1. **限定和** *(bounded sum)*：

$$\sum_{y<z} f(\vec{x},y) = f(\vec{x},0) + f(\vec{x},1) + \cdots + f(\vec{x},z-1)$$

ここで左辺の y は束縛されているものと考える。すなわち $\sum_{y<z} f(\vec{x},y)$ は $n+1$ 個の項 $\vec{x},z$ についての関数である。また、$z=0$ のときは、$\sum_{y<z} f(\vec{x},y) = 0$ とする。

2. **限定積** *(bounded product)*：

$$\prod_{y<z} f(\vec{x},y) = f(\vec{x},0) \cdot f(\vec{x},1) \cdots f(\vec{x},z-1)$$

$z=0$ のときは、$\prod_{y<z} f(\vec{x},y) = 1$ とする。

証明 1. 次のように（h をステップ関数として）原始再帰法を用いて定義することができる：

$$\begin{aligned} h(\vec{x},w_1,w_2) &= \mathsf{plus}(w_2, f(\vec{x},w_1)) \\ \sum_{y<0} f(\vec{x},y) &= 0 \\ \sum_{y<z+1} f(\vec{x},y) &= h(\vec{x},z,\sum_{y<z} f(\vec{x},y)) \end{aligned}$$

第一式と第三式を組み合わせると、

$$\begin{aligned} \sum_{y<z+1} f(\vec{x},y) &= h(\vec{x},z,\sum_{y<z} f(\vec{x},y)) \\ &= \mathsf{plus}(\sum_{y<z} f(\vec{x},y), f(\vec{x},z)) \end{aligned}$$

となる。これと第二式から、どんな $\vec{n},m \in \mathbb{N}$ についても

$$\sum_{y<m} f(\vec{n},y) = f(\vec{n},0) + f(\vec{n},1) + \cdots + f(\vec{n},m-1)$$

となることがわかる。
2. 練習問題とする。 ■

練習問題 3.9

1. $\sum_{y<3} \mathsf{suc}(y) = 6$ となることを確かめよ。
2. $\prod_{y<2} \mathsf{square}(y+1) = 4$ となることを確かめよ。

3.2 原始再帰的関係

定義 3.10 原始再帰的関係 *(primitive recursive relation)* とは、特性関数 χ_R が原始再帰的であるような関係 $R \subseteq \mathbb{N}^k$ のことである。

命題 3.11 大小関係 $x < y$、同値関係 $x = y$ は原始再帰的である。

証明 $x < y$ の特性関数は $\mathsf{pos?}(y \dot{-} x)$ に他ならない。すなわち、$\chi_<(x, y) = \mathsf{pos?}(y \dot{-} x)$ である。実際、任意の $n, m \in \mathbb{N}$ について、

$$\begin{aligned} \mathsf{pos?}(m \dot{-} n) &= 1 \quad (n < m \text{ のとき}) \\ &= 0 \quad (n \not< m \text{ のとき}) \end{aligned}$$

が成り立つ。関数合成により $\mathsf{pos?}(y \dot{-} x)$ は原始再帰的関数なので、定義により $x < y$ は原始再帰的関係である。また、$x = y$ の特性関数は $\mathsf{zero?}(x \dot{-} y) \cdot \mathsf{zero?}(y \dot{-} x)$ と書ける（練習問題：このことを確かめよ）。これは原始再帰的関数なので、$x = y$ は原始再帰的関係である。 ■

k 項原始再帰的関係 R と自然数 $n_1, \ldots, n_k \in \mathbb{N}$ が与えられたとき、$R(n_1, \ldots, n_k)$ が成り立つかどうかは機械的な手続により有限時間で判定可能である（このことを関係 R は**決定可能である** (decidable) という）。そのためには、R の特性関数 χ_R について、$\chi_R(n_1, \ldots, n_k)$ の値を求めればよい。χ_R は原始再帰的関数であり、ゆえに $\chi_R(n_1, \ldots, n_k)$ の値は常に有限時間で計算することができる。もしも値 1 が得られれば、$R(n_1, \ldots, n_k)$ は成り立つし、そうでなければ $R(n_1, \ldots, n_k)$ は成り立たない。

次の命題は、原始再帰的関係と原始再帰的関数を（自由）合成して得られる関係は原始再帰的になることを示している：

命題 3.12 原始再帰的関係 $R \subseteq \mathbb{N}^{k+1}$、原始再帰的関数 $f : \mathbb{N}^l \longrightarrow \mathbb{N}$ が与えられたとき、

$$R'(\vec{x}, \vec{y}) \text{ が成り立つ} \iff R(\vec{x}, f(\vec{y})) \text{ が成り立つ}$$

により定義される関係 $R' \subseteq \mathbb{N}^{k+l}$ は原始再帰的である。

証明 R' の特性関数は自由合成により、$\chi_{R'}(\vec{x}, \vec{y}) = \chi_R(\vec{x}, f(\vec{y}))$ と定義できる。ゆえに原始再帰的。 ■

この命題により、$x + y < z$ や $x + 4 = 2 \cdot x$ などは原始再帰的関係であることがわかる。なぜならば、これらは原始再帰的関係 $<, =$ と原始再帰的関数 $+, \cdot$ を合成して得られる関係だからである。

命題 3.13 (否定・連言・選言) 関係 $R, S \subseteq \mathbb{N}^k$ が原始再帰的のとき、次のように定義される関係 $\neg R$, $R \wedge S$, $R \vee S$ も原始再帰的である：

$$\begin{aligned} \neg R(\vec{x}) \text{ が成り立つ} &\iff R(\vec{x}) \text{ が成り立たない。} \\ R \wedge S(\vec{x}) \text{ が成り立つ} &\iff R(\vec{x}), S(\vec{x}) \text{ が共に成り立たつ。} \\ R \vee S(\vec{x}) \text{ が成り立つ} &\iff R(\vec{x}), S(\vec{x}) \text{ のどちらかが成り立たつ。} \end{aligned}$$

証明　$\neg R$、$R \wedge S$, $R \vee S$ の特性関数は、それぞれ

$$\begin{aligned}
\chi_{\neg R}(\vec{x}) &= \mathsf{zero?}(\chi_R(\vec{x})) \\
\chi_{R\wedge S}(\vec{x}) &= \chi_R(\vec{x}) \cdot \chi_S(\vec{x}) \\
\chi_{R\vee S}(\vec{x}) &= \mathsf{pos?}(\chi_R(\vec{x}) + \chi_S(\vec{x}))
\end{aligned}$$

と定義できる。ゆえに原始再帰的である。 ■

上の命題により、関係 $x \leq y$ が原始再帰的であることがわかる。なぜならば、$x \leq y$ は $x < y \vee x = y$ と同値だからである。

以下、$x \neq y$ は $\neg(x = y)$ の略記とし、$R \to S$, $R \leftrightarrow S$ はそれぞれ $\neg R \vee S$, $(R \to S) \wedge (S \to R)$ の略記とする。R, S が原始再帰的ならば、これらの関係も全て原始再帰的である。

練習問題 3.14 *3* 項関係「x、y、z は互いに異なる」が原始再帰的関係であることを示せ。

命題 3.15 (限定量化) 関係 $R(\vec{x}, y) \subseteq \mathbb{N}^{k+1}$ が原始再帰的のとき、次のように定義される $k+1$ 項関係 $\forall y < z.R(\vec{x}, y)$, $\exists y < z.R(\vec{x}, y)$ も原始再帰的である：$\vec{n}, m \in \mathbb{N}$ とするとき、

$$\begin{aligned}
\forall y < m.R(\vec{n}, y) \text{ が成り立つ} &\iff \text{どんな } i < m \text{ についても } R(\vec{n}, i) \text{ が成り立つ。} \\
\exists y < m.R(\vec{n}, y) \text{ が成り立つ} &\iff \text{ある } i < m \text{ について } R(\vec{n}, i) \text{ が成り立つ。}
\end{aligned}$$

ここで変数 y は束縛されているものと考える。すなわち $\forall y < z.R(\vec{x}, y)$, $\exists y < z.R(\vec{x}, y)$ は $k+1$ 個の項 $\vec{x}, z$ についての関係である。

証明　$\forall y < z.R(\vec{x}, y)$, $\exists y < z.R(\vec{x}, y)$ の特性関数はそれぞれ

$$\begin{aligned}
\chi_{\forall y<z.R(\vec{x},y)}(\vec{x}, z) &= \prod_{y<z} \chi_R(\vec{x}, y) \\
\chi_{\exists y<z.R(\vec{x},y)}(\vec{x}, z) &= \mathsf{pos?}(\sum_{y<z} \chi_R(\vec{x}, y))
\end{aligned}$$

と書ける。 ■

原始再帰的関係 $R(\vec{x}, y) \subseteq \mathbb{N}^{k+1}$ が与えられたとき、上と同様にして、$k+1$ 項関係 $\forall y \leq z.R(\vec{x}, y)$, $\exists y \leq z.R(\vec{x}, y)$ を定義することができる：$\vec{n}, m \in \mathbb{N}$ とするとき、

$$\begin{aligned}
\forall y \leq m.R(\vec{n}, y) \text{ が成り立つ} &\iff \text{どんな } i \leq m \text{ についても } R(\vec{n}, i) \text{ が成り立つ。} \\
\exists y \leq m.R(\vec{n}, y) \text{ が成り立つ} &\iff \text{ある } i \leq m \text{ について } R(\vec{n}, i) \text{ が成り立つ。}
\end{aligned}$$

$\forall y \leq z.R(\vec{x}, y)$, $\exists y \leq z.R(\vec{x}, y)$ は、それぞれ $(\forall y < z.R(\vec{x}, y)) \wedge R(\vec{x}, z)$, $(\exists y < z.R(\vec{x}, y)) \vee R(\vec{x}, z)$ と同値なので、R が原始再帰的のときにはやはり原始再帰的である。$\forall y < z$, $\exists y < z$, $\forall y \leq z$, $\exists y \leq z$ を**限定量化子** (bounded quantifiers) という。

以上により、原始再帰的関数・関係と $=, <, \neg, \wedge, \vee$ 及び限定量化子を用いて記述できる関係は全て原始再帰的であることがわかった。このことから、多くの数論的な性質・関係が原始再帰的であることがわかる：

命題 3.16 次の関係（性質）は原始再帰的である：

1. $\mathsf{even}(x)$：x は偶数である。
2. $\mathsf{div}(x,y)$：x は y の約数である。
3. $\mathsf{prime}(x)$：x は素数である。

証明 1. $\mathsf{even}(x)$ は $\exists y < x(x = 2y)$ と定義できる。ゆえに原始再帰的。
2. $\mathsf{div}(x,y) \Longleftrightarrow \exists z \leq y(x \cdot z = y)$。
3. $\mathsf{prime}(x) \Longleftrightarrow 2 \leq x \wedge \neg\exists y < x(y \neq 1 \wedge \mathsf{div}(y,x))$。 ■

練習問題 3.17 次の関係が原始再帰的であることを示せ。

1. $\mathsf{cd}(x,y,z)$ ： x は y と z の公約数である。
2. $\mathsf{cm}(x,y,z)$ ： x は y と z の公倍数である。
3. $\mathsf{rp}(x,y)$ ： x と y は互いに素である（1 以外の公約数を持たない）。
4. $\mathsf{mersenne}(x)$ ：x はメルセンヌ数である（x は $2^n - 1$ の形の素数である）。

原始再帰的関係は、原子再帰的関数を構成するときにも利用できる。そのような構成法を二つ挙げておく。

命題 3.18 (場合分け) 原始再帰的関数 g、h、原始再帰的関係 R が与えられたとき、次のように定義される関数 f も原始再帰的である。

$$\begin{aligned} f(\vec{x}) &= g(\vec{x}) \ (R(\vec{x}) \text{ が成り立つとき}) \\ &= h(\vec{x}) \ (R(\vec{x}) \text{ が成り立たないとき}) \end{aligned}$$

証明 $f(\vec{x}) = g(\vec{x}) \cdot \chi_R(\vec{x}) + h(\vec{x}) \cdot \chi_{\neg R}(\vec{x})$ より。 ■

このような定義を**場合分けによる定義** (definition by cases) という。例えば、次のように定義される関数 f は原始再帰的である：

$$\begin{aligned} f(x) &= x^2 + 1 \qquad (\mathsf{even}(x) \text{ が成り立つとき}) \\ &= x^2 \qquad\quad (\text{そうでないとき}) \end{aligned}$$

命題 3.19 (限定最小化) 関係 $R(\vec{x},y) \subseteq \mathbb{N}^{k+1}$ が原始再帰的のとき、次のように定義される $k+1$ 項関数 $\mu y < z.R(\vec{x},y)$ は原始再帰的関数である：$\vec{n}, m \in \mathbb{N}$ とするとき、

$$\mu y < m.R(\vec{n},y) = k < m \text{ かつ } R(\vec{n},k) \text{ を満たす最小の } k$$

ただしそのような k が存在しないときには $\mu y < m.R(\vec{n},y) = m$ とする。

ここでも y はやはり束縛されているものと考える。すなわち $\mu y < z.R(\vec{x}, y)$ は項 $\vec{x}, z$ についての関数である。このような関数の構成法を**限定最小化** (bounded minimization) という。

証明　次のように場合分けと原始再帰法を組み合わせることにより定義できる。

$$\begin{array}{rcll} \mu y < 0.R(\vec{x}, y) & = & 0 & \\ \mu y < z+1.R(\vec{x}, y) & = & \mu y < z.R(\vec{x}, y) & (\exists y \leq z.R(\vec{x}, y) \text{ が成り立つとき}) \\ & = & z+1 & (\text{そうでないとき}) \end{array}$$

■

例えば、x と y の最大公約数を返す関数 $\mathsf{gcd}(x, y)$ は原始再帰的である。なぜならば、$\mathsf{gcd}(x, y)$ は次のように定義できるからである：

$$\mathsf{gcd}(x, y) \quad = \quad \mu z \leq y.(\mathsf{cd}(x, y, z) \wedge \neg \exists w \leq y(z < w \wedge \mathsf{cd}(x, y, w)))$$

練習問題 3.20　*1.* x と y の最小公倍数を返す関数 $\mathsf{lcm}(x, y)$ が原始再帰的であることを示せ。

2. $\mu y < z.R(\vec{x}, y) = \sum_{w<z} \prod_{y \leq w} \chi_{\neg R}(\vec{x}, y)$ となることを確かめよ。（ゆえに限定最小化は $\sum$ と $\prod$ を用いても定義できる。）

3. 関係 $R(\vec{x}, y) \subseteq \mathbb{N}^{k+1}$ が原始再帰的のとき、次のように定義される $k+1$ 項関数 $\sharp_{y<z} R(\vec{x}, y)$ が原始再帰的関数であることを示せ：$\vec{n}, m \in \mathbb{N}^k$ とするとき、

$$\sharp_{y<m} R(\vec{n}, y) \quad = \quad k < m \text{ かつ } R(\vec{n}, k) \text{ を満たす } k \text{ の個数}$$

（**限定数え上げ** *(bounded counting)*）

限定最小化を用いることにより、後々重要になる次の命題が証明できる。

命題 3.21　n 番目の素数を p_n により表すことにする。このとき p_x は原始再帰的関数である。

ただし、0 番目から数えていくことにする。例えば $\mathsf{p}_0 = 2$、$\mathsf{p}_1 = 3$、$\mathsf{p}_2 = 5$ である。

証明　$\mathsf{p}_n! + 1$ の最小の自明でない約数（1 以外の約数）を m とすると、$\mathsf{p}_n < m \leq \mathsf{p}_n! + 1$ が成り立つ。最初の不等号が成り立つのは、$\mathsf{p}_n! + 1$ は 2 以上 p_n 以下のどのような数であっても必ず 1 余り、すなわち、$\mathsf{p}_n! + 1$ の自明でない約数は p_n 以下には存在せず、ゆえに m は p_n 以下ではありえないからである。二番目の不等号は自明である。また、m は素数である。なぜならばどのような数についても、その最小の自明でない約数は常に素数だからである。このことから、p_n の次の素数は、必ず $\mathsf{p}_n! + 1$ までの範囲の中に見つかることがわかる。

適切な上界 (upper bound) が得られたので限定最小化を用いることができ、p_x は次のように定義することができる：

$$\begin{aligned} \mathsf{p}_0 &= 2 \\ \mathsf{p}_{x+1} &= \mu y \leq \mathsf{p}_x! + 1.(\mathsf{p}_x < y \wedge \mathsf{prime}(y)) \end{aligned}$$

■

素因数分解の一意性を用いれば、自然数の列を一つの自然数によりコード化することができる。自然数の列 $n_0, \ldots, n_k$ が与えられたとき、

$$\langle n_0, \ldots, n_k \rangle = \mathsf{p}_0^{n_1+1} \cdots \mathsf{p}_k^{n_k+1}$$

と定義する。これは $k+1$ 項の原始再帰的関数である。自然数 x が上のようなかたちで何らかの列を表すとき、その i 番目の要素は

$$(x)_i = \mu y < x(\neg \mathsf{div}(\mathsf{p}_i^{y+2}, x))$$

により取り出すことができる。コード x の長さは

$$\mathsf{leng}(x) = \mu i < x(\neg \mathsf{div}(\mathsf{p}_i, x))$$

と表すことができ、また自然数 x が列のコードになるための条件は

$$\mathsf{seq}(x) \Longleftrightarrow \forall i < x(\mathsf{div}(\mathsf{p}_i, x) \rightarrow i < \mathsf{leng}(x))$$

と表すことができる。これらは全て原始再帰的である。

3.3 原始再帰的関数の限界

これまではどのようなタイプの関数が原始再帰的と見なせるかという、いわば原始再帰的関数の質的な側面に着目してきたが、ここでは視点を変えて、どれくらい大きな関数が原始再帰的であるかという、原始再帰的関数の量的側面について考えてみる。関数の大きさは “支配する” という関係によって与えることができる。1 項関数 f が g を支配するとは、十分に大きな数 N をとれば、それより大きな全ての数 $x > N$ について $g(x) \leq f(x)$ となることとする。例えば、二次関数 x^2 は全ての一次関数 $ax+b$ を支配する（ここで a, b は定数）。同様に三次関数 x^3 は全ての二次関数を支配し、べき乗関数 2^x は全ての多項式を支配する。超指数関数 2^{2^x} は全ての指数関数を支配する、といった具合である。

ここでは原始再帰的関数の列 $\Upsilon_0, \Upsilon_1, \Upsilon_2, \ldots$ を構成し、どのような原始再帰的関数 f もある Υ_n により支配されることを示す。一方、関数列 $\Upsilon_0, \Upsilon_1, \Upsilon_2 \ldots$ を “対角化” することにより、明らかに計算可能でありながらどんな Υ_n によっても支配されない関数 $\Upsilon_x(x)$ を構成することができる。そのような関数は原始再帰的ではありえない。ゆえに原始再帰的関数のクラスは全ての計算可能な関数を覆い尽くしてはいないことがわかる。

命題 3.22 *1* 項原始再帰的関数 $f(x)$ が与えられたとき、二項原始再帰的関数 $f^{(y)}(x)$ を次のように定義する：

$$f^{(y)}(x) = \underbrace{f \cdots f}_{y\ times}(x)$$

ただし $f^{(0)}(x) = x$ とする。このとき $f^{(y)}(x)$ は原始再帰的関数である。

この関数構成法は原始再帰法の特別な場合であり、**反復法**（iteration）と呼ばれる。

証明

$$\begin{aligned} f^{(0)}(x) &= x \\ f^{(y+1)}(x) &= f(f^{(y)}(x)) \end{aligned}$$

■

定義 3.23 *1* 項原始再帰的関数の列 $\Upsilon_0, \Upsilon_1, \Upsilon_2 \ldots$ を次のように帰納的に定義する：

$$\begin{aligned} \Upsilon_0(x) &= \mathsf{suc}(x) \\ \Upsilon_{n+1}(x) &= \Upsilon_n^{(x)}(x) \end{aligned}$$

例えば、

$$\begin{aligned} \Upsilon_1(x) &= \Upsilon_0^{(x)}(x) = \mathsf{suc}^{(x)}(x) = 2x \\ \Upsilon_2(x) &= \Upsilon_1^{(x)}(x) = 2^{(x)}(x) = 2^x \cdot x \leq 3^x \\ \Upsilon_3(x) &= \Upsilon_2^{(x)}(x) \leq 3^{\cdot^{\cdot^{\cdot^{3^x}}}} \Big\} x\ times \end{aligned}$$

となる。

補題 3.24 $n, m, x, y \in \mathbb{N}$ とする。

1. $x \leq \Upsilon_n(x)$. さらに $1 \leq x$ ならば $x < \Upsilon_n(x)$.
2. $1 \leq n < m$ ならば $\Upsilon_n(x) \leq \Upsilon_m(x)$. さらに $2 \leq x$ ならば $\Upsilon_n(x) < \Upsilon_m(x)$.
3. $x \leq y$ ならば $\Upsilon_n(x) \leq \Upsilon_n(y)$.
4. $x_1 \leq y,\ x_2 \leq y$ ならば $\Upsilon_n^{(x_1)}(x_2) \leq \Upsilon_{n+1}(y)$.

証明 1. n についての帰納法による。まず、$x < \mathsf{suc}(x) = \Upsilon_0(x)$。つぎに $x \leq \Upsilon_n(x)$ が成り立つと仮定して（帰納法の仮定）、$x \leq \Upsilon_{n+1}(x)$ が成り立つことを示す。実際どんな x についても、帰納法の仮定を x 回用いれば

$$x \leq \Upsilon_n(x) \leq \Upsilon_n(\Upsilon_n(x)) \leq \cdots \leq \underbrace{\Upsilon_n \cdots \Upsilon_n}_{x\ times}(x) \leq \Upsilon_n^{(x)}(x) = \Upsilon_{n+1}(x).$$

よってどんな n についても $x \leq \Upsilon_n(x)$ である。$1 \leq x$ のとき $x < \Upsilon_n(x)$ となることも同様にして確かめることができる。
2. $m = n + 1$ の場合を示せば十分。$x = 0$ のときは $\Upsilon_n(x) = 0 = \Upsilon_{n+1}(x)$. $1 \leq x$ のときは上記 1 より

$$\Upsilon_n(x) \leq \Upsilon_n^{(x)}(x) = \Upsilon_{n+1}(x).$$

また $2 \leq x$ のときは $1 \leq \Upsilon_n(x)$ なので 1 の後半の主張より

$$\Upsilon_n(x) < \Upsilon_n(\Upsilon_n((x)) \leq \Upsilon_n^{(x)}(x) = \Upsilon_{n+1}(x).$$

3. n についての帰納法による。まず、$\Upsilon_0(x) = \mathsf{suc}(x) \leq \mathsf{suc}(y) = \Upsilon_0(y)$. 次に $\Upsilon_n(x) \leq \Upsilon_n(y)$ が成り立つと仮定して（帰納法の仮定）、$\Upsilon_{n+1}(x) \leq \Upsilon_{n+1}(y)$ が成り立つことを示す。実際上記 1 と帰納法の仮定により、どんな x についても

$$\begin{aligned}
\Upsilon_{n+1}(x) &= \Upsilon_n^{(x)}(x) \\
&= \underbrace{\Upsilon_n \cdots \Upsilon_n}_{x\ times}(x) \\
&\leq \underbrace{\Upsilon_n \cdots \Upsilon_n}_{x\ times}(y) \quad \text{（帰納法の仮定を } x \text{ 回用いて）} \\
&\leq \underbrace{\Upsilon_n \cdots \Upsilon_n}_{y\ times}(y) \quad \text{（上記 1 より）} \\
&= \Upsilon_n^{(y)}(y) = \Upsilon_{n+1}(y)
\end{aligned}$$

が成り立つ。
4. 上記 1 と 3 より。 ■

定義 3.25 *f を 1 項関数とする。f が 1 項関数 g を* **支配する** *(dominates)* とは、ある $N \in \mathbb{N}$ が存在し、全ての $x > N$ について $g(x) \leq f(x)$ が成り立つこととする。同様に、f が n 項関数 g を支配するとは、ある $N \in \mathbb{N}$ が存在し、全ての $\vec{x} = x_1, \ldots, x_n > N$ について $g(\vec{x}) \leq f(\mathsf{max}(\vec{x}))$ が成り立つこととする。

すなわち、f が g を支配するとは、十分大きなインプットについては常に f のアウトプットは g のアウトプット以上となることに他ならない。

定理 3.26 どんな原始再帰的関数 f もある Υ_n（$n \geq 1$）により支配される。より詳しく言えば、f と Υ_n はどんな $\vec{x} = x_1, \ldots, x_k \geq 2$ についても $f(\vec{x}) \leq \Upsilon_n(\mathsf{max}(\vec{x}))$ を満たす。

証明 f の構成に関する帰納法により証明する。

f が zero, suc, proj のときには明らかに Υ_1 により支配される（実際には Υ_0 で十分）。

f が原始再帰的関数 $g,\ h_1, \ldots, h_m$ から合成により定義されているとする（すなわち $f(\vec{x}) = g(h_1(\vec{x}), \ldots, h_m(\vec{x}))$）。帰納法の仮定及び補題 3.24.2 により、$n$ を十分に大きくとれば、全ての $\vec{x}, \vec{y} \geq 2$ について $g(\vec{y}) \leq \Upsilon_n(\mathsf{max}(\vec{y}))$, $h_1(\vec{x}) \leq \Upsilon_n(\mathsf{max}(\vec{x}))$, $\ldots$, $h_m(\vec{x}) \leq \Upsilon_n(\mathsf{max}(\vec{x}))$ が成り立つ。よって

$$\begin{aligned}
f(\vec{x}) &= g(h_1(\vec{x}), \ldots, h_m(\vec{x})) \\
&\leq \Upsilon_n(\mathsf{max}(h_1(\vec{x}), \ldots, h_m(\vec{x}))) \\
&\leq \Upsilon_n(\mathsf{max}(\Upsilon_n(\mathsf{max}(\vec{x})), \ldots, \Upsilon_n(\mathsf{max}(\vec{x})))) \\
&= \Upsilon_n(\Upsilon_n(\mathsf{max}(\vec{x}))) \\
&= \Upsilon_n^{(2)}(\mathsf{max}(\vec{x})) \\
&\leq \Upsilon_{n+1}(\mathsf{max}(\vec{x}))\ \ (\mathsf{max}(\vec{x}) \geq 2、\text{補題 3.24.4 より})
\end{aligned}$$

f が原始再帰的関数 g, h から原始再帰法により定義されているとする。すなわち、

$$\begin{aligned}
f(\vec{x}, 0) &= g(\vec{x}) \\
f(\vec{x}, y+1) &= h(\vec{x}, y, f(\vec{x}, y)).
\end{aligned}$$

帰納法の仮定及び補題 3.24.2 により、ある $n \geq 1$ があり、全ての $\vec{x}, y, z \geq 2$ について、$g(\vec{x}) \leq \Upsilon_n(\mathsf{max}(\vec{x}))$, $h(\vec{x}, y, z) \leq \Upsilon_n(\mathsf{max}(\vec{x}, y, z))$ が成り立つ。

このとき $f(\vec{x}, m) \leq \Upsilon_n^{(m+1)}(\mathsf{max}(\vec{x}, m))$ となることを m についての数学的帰納法により示す。実際、

$$\begin{aligned}
f(\vec{x}, 0) &= g(\vec{x}) \leq \Upsilon_n(\mathsf{max}(\vec{x})) = \Upsilon_n^{(1)}(\mathsf{max}(\vec{x}, 0)) \\
f(\vec{x}, m+1) &= h(\vec{x}, m, f(\vec{x}, m)) \\
&\leq \Upsilon_n(\mathsf{max}(\vec{x}, m, f(\vec{x}, m))) \\
&\leq \Upsilon_n(\mathsf{max}(\vec{x}, m, \Upsilon_n^{(m+1)}(\mathsf{max}(\vec{x}, m))))\ \ (\text{帰納法の仮定、補題 3.24.3 より}) \\
&= \Upsilon_n(\Upsilon_n^{(m+1)}(\mathsf{max}(\vec{x}, m+1)))\ \ (\text{補題 3.24.1 より}) \\
&= \Upsilon_n^{(m+2)}(\mathsf{max}(\vec{x}, m+1)).
\end{aligned}$$

ゆえに、

$$\begin{aligned} f(\vec{x},y) &\leq \Upsilon_n^{(y+1)}(\mathsf{max}(\vec{x},y)) \\ &= \Upsilon_n(\Upsilon_n^{(y)}(\mathsf{max}(\vec{x},y))) \\ &\leq \Upsilon_n(\Upsilon_{n+1}(\mathsf{max}(\vec{x},y))) \quad \text{（補題 3.24.4 より）} \\ &\leq \Upsilon_{n+1}(\Upsilon_{n+1}(\mathsf{max}(\vec{x},y))) \\ &= \Upsilon_{n+1}^{(2)}(\mathsf{max}(\vec{x},y)) \\ &\leq \Upsilon_{n+2}(\mathsf{max}(\vec{x},y)) \quad \text{（}\mathsf{max}(\vec{x},y)\geq 2\text{、補題 3.24.4 より）} \end{aligned}$$

■

系 3.27 $\Upsilon(x)=\Upsilon_x(x)$ により定義される関数 Υ は原始再帰的ではない。

証明 仮に Υ が原始再帰的であるとすると、定理 3.26 により、Υ はある Υ_n $(n\geq 1)$ により支配されるはずである。とくに $\Upsilon(n+1)=\Upsilon_{n+1}(n+1)\leq\Upsilon_n(n+1)$ となるずであるが、これは補題 3.24.2 $(\Upsilon_n(n+1)<\Upsilon_{n+1}(n+1))$ と矛盾する。 ■

一方、Υ が計算可能であることは明らかである。なぜならば具体的なインプット n が与えられたとき、$\Upsilon(n)=\Upsilon_n(n)$ であり、Υ_n は原始再帰的であるから、$\Upsilon_n(n)$ の値は確かに求めることができる。ゆえにこの Υ は計算可能であっても原始再帰的でない関数の具体例になっている（後ほど Υ は再帰的関数であることが明らかになる)。

ところで上のように関数の列 $\Upsilon_0,\Upsilon_1,\Upsilon_2,\ldots$ から Υ を構成する方法を**対角化** (diagonalization) と言う。これは、$\Upsilon_n(m)$ の値（$n=0,1,\ldots;m=0,1,\ldots$）を表

	0	1	$\cdots$	m	$\cdots$
0	$\Upsilon_0(0)$	$\Upsilon_0(1)$	$\cdots$	$\Upsilon_0(m)$	$\cdots$
1	$\Upsilon_1(0)$	$\Upsilon_1(1)$	$\cdots$	$\Upsilon_1(m)$	$\cdots$
$\vdots$	$\vdots$	$\vdots$		$\vdots$	
n	$\Upsilon_n(0)$	$\Upsilon_n(1)$	$\cdots$	$\Upsilon_n(m)$	$\cdots$
$\vdots$	$\vdots$	$\vdots$		$\vdots$	

で表したとき、$\Upsilon(n)$ の値（$n=0,1,\ldots$）はその対角線に相当するからである。

練習問題 3.28 *(**)* 次のように定義される関数（**アッカーマン関数**）が原始再帰的関数ではないことを証明せよ。

$$\begin{aligned} f(0,y) &= y+1 \\ f(x+1,0) &= f(x,1) \\ f(x+1,y+1) &= f(x,f(x+1,y)). \end{aligned}$$

3.4 コード化の技法

ここでは数の順序対や有限集合、数の有限列など数以外のものを一つの数で表す**コード化**の技法を紹介する。既に 3.2 章で素因数分解の一意性に基づいて数の有限列をコード化する方法について触れたが、ここで導入するのはより能率のよい（したがってより工夫を要する）方法である。これらは後に**算術化**（arithmetization, 論理式や証明などのメタレヴェルの概念を数についての関数や性質として対象レヴェルで表現すること）を行う際に本質的な役割を果たす。まずは原始再帰的関係のクラスよりもはるかに小さい Δ_0 関係のクラスを導入しておこう。

定義 3.29 Δ_0 関係のクラスは次のように定義される：

1. f, g が $0, \mathsf{suc}, +, \cdot$ から自由合成（補題 *3.5*）により得られる関数のとき、$f(\vec{x}) = g(\vec{x})$ は Δ_0 関係である。
2. R, S が Δ_0 関係のとき、$\neg R, R \wedge S, R \vee S$ は Δ_0 関係である。
3. R が Δ_0 関係で f が $0, \mathsf{suc}, +, \cdot$ から自由合成により得られる関数のとき、$\forall y < f(\vec{x}).R(\vec{x}, y)$, $\exists y < f(\vec{x}).R(\vec{x}, y)$ は Δ_0 関係である。ただし y は $\vec{x}$ の中に現れない変数とする。

すなわち Δ_0 関係とは、$0, \mathsf{suc}, +, \cdot, =, \neg, \wedge, \vee$ および限定量化子（および合成）のみを用いて定義することができる関係のことである。

まず最初に、自然数の順序対 (m, n) を一つの自然数でコード化する方法について考える。順序対全体の集合は $\mathbb{N}^2$ であり、これは x 軸、y 軸とも自然数で目盛りづけられた平面上の格子点の集合と一致する。次のような数え上げ方を見てみよう。

$$
\begin{array}{l}
(0,0) \\
(1,0), (0,1) \\
(2,0), (1,1), (0,2) \\
(3,0), (2,1), (1,2), (0,3) \\
(4,0), (3,1), (2,2), (1,3), (0,4) \\
\quad \vdots
\end{array}
$$

このようにすればちょうど一回ずつ、全ての格子点を数え上げることができる。言い換えれば、集合 $\mathbb{N}^2$ を集合 $\mathbb{N}$ へと一対一に対応付けることができる。この数え方において（一番上の行を 0 行目とすれば）、

- (m, n) は $m + n$ 行目に現れる（例えば $(3, 1)$ は 4 行目に現れる）
- i 行目には $i + 1$ 個の格子点が現れる（例えば 4 行目には 5 個の格子点が現れる）

ことに注目すると、格子点 (m, n) は

$$\left(\sum_{i<m+n} i + 1 \right) + n = \frac{(m+n)(m+n-1)}{2} + n$$

番目に数え上げられることがわかる。ゆえに 2 項関数

$$\langle x, y\rangle = \frac{(x+y)(x+y-1)}{2} + y$$

を順序対を単一の自然数でコード化するために用いることができる。これは原始再帰的であり、全単射である。逆関数は

$$\begin{aligned} \pi_1(z) &= \mu x \leq z(\exists y \leq z.z = \langle x, y\rangle) \\ \pi_2(z) &= \mu y \leq z(\exists x \leq z.z = \langle x, y\rangle) \end{aligned}$$

により与えられる。すなわち $\pi_i(\langle n_1, n_2\rangle) = n_i$ $(i = 1, 2)$ である。これらも原始再帰的関数である。しかも次の性質が成り立つ。

命題 3.30 *3 項関係* $\langle x, y\rangle = z$ は Δ_0 である。

証明 $\langle x, y\rangle = z \Longleftrightarrow 2z = (m+n)(m+n-1) + 2n$ より。 ■

練習問題 3.31 *(*)* $\langle x, y\rangle$ が $\mathbb{N}^2$ から $\mathbb{N}$ への全単射であることを証明せよ。

以上により順序対は自然数によりコード化できることがわかった。次に有限集合のコード化について考えよう。以下の手法はクワインとスマリヤンによる。

補題 3.32 次の関係は Δ_0 である。

1. $\mathsf{power}_p(x)$ *:* x は p の累乗である（すなわち、ある n について $x = p^n$）。ただし p は素数であるとする。
2. $y = \mathsf{lstpow}_p(x)$ *:* y は x より大きな p の累乗のうち最小のものである。ただし p は素数であるとする。

たとえば $\mathsf{power}_2(16)$ は成り立つが、$\mathsf{power}_2(18)$ は成り立たない。$16 \leq n < 32$ のとき、$32 = \mathsf{lstpow}_2(n)$ である。自然数 n を p 進法で書くときに必要な桁数を k とすると、$\mathsf{lstpow}_p(n)$ は p 進法で $1\underbrace{0\cdots0}_{k}$ と書ける。

証明 1. p が素数のとき、x が p の累乗であるための必要十分条件は x の全ての自明でない約数（1 以外の約数）が p で割り切れることである。ゆえに:

$$\mathsf{power}_p(x) \Longleftrightarrow \forall z \leq x((\mathsf{div}(z, x) \wedge z \neq 1) \rightarrow \mathsf{div}(p, z)).$$

(div が Δ_0 関係であることは命題 3.16 の証明から明らかである。)
2. 次の通り:

$$y = \mathsf{lstpow}_p(x) \Longleftrightarrow (y > x \wedge \mathsf{power}_p(y)) \wedge \neg\exists z < y(z > x \wedge \mathsf{power}_p(z)).$$

■

「x と y を p 進法で書いて x, y の順に繋げることにより得られる数」を $x *_p y$ により表すことにする。たとえば、$123 *_{10} 4567 = 1234567$ であり、$3 *_2 2 = 14$ である（二進数で書くと 3 は 11、2 は 10、14 は 1110 である)。

補題 3.33 p が素数のとき、3 項関係 $x *_p y = z$ は Δ_0 関係である。

証明

$$\begin{aligned} x *_p y = z &\iff x \cdot \mathsf{lstpow}_p(y) + y = z \\ &\iff \exists w \leq z. w = \mathsf{lstpow}_p(y) \wedge x \cdot w + y = z. \end{aligned}$$

■

x と y を p 進法で書いたとき、「x が y の始切片である」ことを $\mathsf{initseg}_p(x, y)$、「x が y の終切片である」ことを $\mathsf{endseg}_p(x, y)$、「x が y の部分である」ことを $\mathsf{part}_p(x, y)$ と書いて表す。例えば、$\mathsf{initseg}_{10}(123, 123456789)$、$\mathsf{endseg}_{10}(789, 123456789)$、$\mathsf{part}_{10}(456, 123456789)$ が成り立つ。

補題 3.34 p が素数のとき、$\mathsf{initseg}_p(x, y)$、$\mathsf{endseg}_p(x, y)$、$\mathsf{part}_p(x, y)$ は Δ_0 関係である。

証明

$$\begin{aligned} \mathsf{initseg}_p(x, y) &\iff x = y \vee (x \neq 0 \wedge \exists z < y. x *_p z = y) \\ \mathsf{endseg}_p(x, y) &\iff x = y \vee (x \neq 0 \wedge \exists z < y. z *_p x = y) \\ \mathsf{part}_p(x, y) &\iff \exists z < y. \mathsf{initseg}_p(z, y) \wedge \mathsf{endseg}_p(x, z). \end{aligned}$$

■

以下では素数 $p > 2$ を固定し（例えば $p = 7$ とする）、$\mathsf{part}_p(x, y)$ を単に $\mathsf{part}(x, y)$ と書く。また $x *_p y$ を単に xy と書く。p 進法で書いたとき $2111 \cdots 12$ の形になる数を**区切り数**と呼ぶことにする。

補題 3.35 次の関係は Δ_0 である。

1. $\mathsf{1seq}(x)$: x は p 進法で書いたとき $1 \cdots 1$ の形になる。
2. $\mathsf{delim}(x)$: x は区切り数である。
3. $\mathsf{maxdelim}(x, y)$: x は y を p 進法で書いたときその中に現れる最大の区切り数である。

証明

$$\begin{aligned} \mathsf{1seq}(x) &\iff x \neq 0 \wedge \forall y \leq x (\mathsf{part}(y, x) \wedge y < p \to y = 1) \\ \mathsf{delim}(x) &\iff \exists z < x. (x = 2z2 \wedge \mathsf{1seq}(z)) \\ \mathsf{maxdelim}(x, y) &\iff \mathsf{delim}(x) \wedge \neg \exists z \leq y (\mathsf{delim}(z) \wedge x < z) \end{aligned}$$

■

さて、数の有限集合 $A = \{a_0, \ldots, a_n\}$ が与えられたとする。どの a_i の部分でもない区切り数を一つ取りそれを l とするとき、$la_0la_1l \cdots la_nl$ の形の自然数を A の**コード**ということにする。A のコードは一意には定まらない。それは区切り数 l の取り方に依存するからである。しかしそれでも、「x は $\{a_0, \ldots, a_n\}$ のコードである」という関係を考えることには意味があり、これが原始再帰的関係であることは容易に確かめることができる。このような状況の下で、次の命題が成り立つ。

命題 3.36 次の性質を持つ Δ_0 関係 $\in$ が存在する: どんな有限集合 $A = \{a_0, \ldots, a_n\}$ についてもある y が存在し、

$$x \in y \Longleftrightarrow x \text{ は } A \text{ の要素である。}$$

また $a_0, \ldots, a_n < y$ である。

ここで y としては A の（任意の）コードが想定されている。

証明 $x \in y \Longleftrightarrow \exists z < y.\mathsf{maxdelim}(z, y) \wedge \mathsf{part}(zxz, y)$ と定義すればよい。いま、l をどの a_i の部分でもない区切り数とするとき、それを用いて構成できる A の集合コード $la_0la_1l\cdots la_nl$ を y として取る。すると、l は y に含まれる最大の区切り数であり、lxl が y の部分であることと x が A の要素であることは一致する。すなわち、$x \in y$ であることと x が A の要素であることは一致する。$a_0, \ldots, a_n < y$ であることは明らかである。 ■

順序対のコード化と有限集合のコード化が与えられれば有限列のコード化は容易である。数列 $a_0, \ldots, a_n$ は集合 $\{\langle 0, a_0\rangle, \langle 1, a_1\rangle, \ldots, \langle n, a_n\rangle\}$ であいまい性なく表現できるからである。次の定理はいわゆる β **関数**の存在を主張している。

定理 3.37 次の性質を満たす *2* 項原始再帰的関数 β が存在する。

1. どんな自然数の列 $a_0, \ldots, a_n$ に対してもある自然数 w が存在し、

   $$\beta(w, 0) = a_0,\ \beta(w, 1) = a_1,\ \ldots,\ \beta(w, n) = a_n$$

   が成り立つ。また、$a_0, \ldots, a_n < w$ である。

2. *3* 項関係 $\beta(w, x) = y$ は Δ_0 関係である。

証明

$$\beta(w, x) = \mu y < w.\langle x, y\rangle \in w$$

と定義すればよい。w を $\{\langle 0, a_0\rangle, \ldots, \langle n, a_n\rangle\}$ のコードとすれば、明らかに $\beta(w, 0) = a_0, \beta(w, 1) = a_1, \ldots, \beta(w, n) = a_n$ が成り立つ。

また、関係 $\beta(w, x) = y$ は

$$(\langle x, y\rangle \in w \wedge \forall z < y.\langle x, z\rangle \notin w) \vee (\neg\exists z \leq w(\langle x, z\rangle \in w) \wedge y = w)$$

と書けるので Δ_0 である。 ■

β 関数を最初に明示的に導入したのはゲーデル (1931) である。彼は中国剰余定理という数論の基本定理の一つを用いて β 関数を構成した。本節の構成法はクワイン、スマリヤンによる。

4 再帰的関数

4.1 再帰的関数

前章までに取り扱ってきた原始再帰的関数の定義を拡張することにより、再帰的関数のクラスが得られる。後で説明するように、再帰的関数は「計算可能である」という直感的概念を数学的に定式化したものと考えることができる（チャーチのテーゼ)。

定義 4.1 再帰的関数 *(recursive functions)* の集合は次のように帰納的に定義される。

1. ゼロ関数、射影関数、後続者関数は再帰的である。
2. 再帰的関数から合成により得られる関数は再帰的である。
3. 再帰的関数から原始再帰法により得られる関数は再帰的である。
4. **最小化**（*minimization*）：g が $n+1$ 項再帰的関数であり、

   $$\forall x_1 \cdots \forall x_n \exists y (g(x_1, \ldots, x_n, y) = 0)$$

   を満たすとする（これを**実効性条件**（*effectiveness condition*）と呼ぶ)。$g(x_1, \ldots, x_n, y) = 0$ を満たす最小の y を $\mu y(g(x_1, \ldots, x_n, y) = 0)$ と書いて表す。このとき

   $$f(x_1, \ldots, x_n) = \mu y(g(x_1, \ldots, x_n, y) = 0)$$

   により定義される関数 f は再帰的である。

   再帰的関係 *(recursive relations)* とは、特性関数が再帰的であるような関係のことである。

最小化による関数の構成法を次のように言い換えてもよいことはただちにわかる。

- g が $n+1$ 項再帰的関係であり、$\forall x_1 \cdots \forall x_n \exists y (R(x_1, \ldots, x_n, y))$ を満たすとする（これを**実効性条件**と呼ぶ)。$R(x_1, \ldots, x_n, y)$ を満たす最小の y を $\mu y(R(x_1, \ldots, x_n, y))$ と書いて表す。このとき

  $$f(x_1, \ldots, x_n) = \mu y(R(x_1, \ldots, x_n, y))$$

  により定義される関数 f は再帰的である。

関係 R が決定可能であり、かつ実効性条件を満たすときには、上のように最小化により定義される f が計算可能であることはただちにわかる。実際、インプット $\vec{m} = m_1, \ldots, m_n \in \mathbb{N}$ が与えられたとき $f(\vec{m})$ の値を求めるには次のようにすればよい。

1. $R(\vec{m}, 0)$ が成り立つかどうかを判定する（R は決定可能であると仮定しているので、このことは有限時間内に遂行できる)。もしも成り立つならば、$f(\vec{m}) = 0$ である。
2. さもなくば、$R(\vec{m}, 1)$ が成り立つかどうかを判定する。もしも成り立つならば、$f(\vec{m}) = 1$ である。さもなくば、$R(\vec{m}, 2)$ が成り立つかどうかを判定する。

3. 以下同様に、アウトプットが得られるまで $R(\vec{m},3), R(\vec{m},4), R(\vec{m},5), \ldots$ と続ける。

R は実効性条件を満たすので、ある k が存在し、$R(\vec{m},k)$ が成り立つ。ゆえに上のプロセスは必ずいつかは停止し、$f(\vec{m})$ のアウトプットが得られるはずである。しかし一般には、上のプロセスがいつ停止するのかは全くわからず、実行時間の予想は立てられない。この点が原始再帰法との大きな違いである。

上の再帰的関数の定義は原始再帰法と最小化を両方用いているが、実は次の定理が述べるように、原始再帰法は初期関数が十分に多くある場合には最小化によってシミュレートできる。このことは最小化がいかに強力な操作であるかを示している。

定理 4.2 次のように定義される再帰的' 関数のクラスは再帰的関数のクラスと一致する。

1. ゼロ関数、射影関数、後続者関数、$+, \cdot$ および $\chi_=$ は再帰的' である。
2. 再帰的' 関数から合成および最小化により得られる関数は再帰的' である。

証明 再帰的' 関数が再帰的であることは明らかである。逆を示すために、まずは Δ_0 関係が再帰的' であることを証明しておく。

1. 関係 $f(\vec{x}) = g(\vec{x})$ が再帰的' であることは明らか（ここで f, g は $0, \mathsf{suc}, +, \cdot$ から自由合成により構成される関数)。
2. R が再帰的' のとき $\neg R$ も再帰的' である。実際、$\neg R$ の特性関数は $\chi_{\neg R}(\vec{x}) = \chi_=(\chi_R(\vec{x}), 0)$ と定義することができ、$\chi_=$ は定義により再帰的' だからである。
3. R が再帰的' のとき $R \vee S$, $R \wedge S$ も再帰的' である。練習問題とする。
4. $R(\vec{x}, z)$ が再帰的' のとき限定最小化により得られる関数 $f(\vec{x}, y) = \mu z < y.R(\vec{x}, z)$ も再帰的' である。実際、$f(\vec{x}, y) = \mu z.(R(\vec{x}, z) \vee z = y)$ と定義できる。
5. $R(\vec{x}, z)$ が再帰的' のとき $\exists z < y.R(\vec{x}, z)$ も再帰的' である。実際、$\exists z < y.R(\vec{x}, z) \iff (\mu z < y.R(\vec{x}, z)) \neq y$ と定義でき、$\neq$ は再帰的' であり、また限定最小化は再帰的' であることを保存することから明らかである。$\forall z < x.R(\vec{x}, z)$ は $\neg$ と $\exists z < x$ を用いて定義できるから再帰的' である。

これで全ての Δ_0 関係が再帰的' であることがわかった。特に 3.4 章の β 関数は再帰的' である。

最後に再帰的関数がすべて再帰的' であることを示そう。それには g, h が再帰的' 関数ならば

$$\begin{aligned} f(\vec{x}, 0) &= g(\vec{x}) \\ f(\vec{x}, y+1) &= h(\vec{x}, y, f(\vec{x}, y)). \end{aligned}$$

により定義される f も再帰的' であることを示せばよい。まず、どんな $\vec{x}, y$ についても有限列 $f(\vec{x}, 0), \ldots, f(\vec{x}, y)$ のコードを返す関数 $t(\vec{x}, y)$ は再帰的' であることに注意する。実際、t は β 関数と最小化を用いて

$$t(\vec{x}, y) = \mu z(\beta(z, 0) = g(\vec{x}) \wedge \forall i < y.\beta(z, i+1) = h(\vec{x}, i, \beta(z, i)))$$

と定義することができる。ここで関係 $\beta(z,0) = g(\vec{x}) \wedge \forall i < y.\beta(z,i+1) = h(\vec{x},i,\beta(z,i)))$ が実効性条件を満たすこと、すなわちどんな $\vec{x}, y$ が与えられてもこの関係を満たす z が存在することは、定理 3.37 により保証されている。

この t を用いれば、目標の関数 f は

$$f(\vec{x},y) = \beta(t(\vec{x},y),y)$$

と定義できる。よって f は再帰的' である。 ■

4.2 部分再帰的関数

再帰的関数は「計算可能な」関数という直感的概念を数学的に定式化したものである。ここで計算可能な関数 f とは、すなわちインプットが与えられたときにアウトプットを計算するための具体的な手続が存在し、その手続は常に有限時間で実行できるというようなもののことだと思ってよい。これを少し弱めて次のような直感的概念にも数学的定式化を与えておくと便利である :「f にはインプットが与えられたときアウトプットを計算するための具体的な手続があるのだが、その手続は常に有限時間で実行できるわけではなく、ときには計算がいつまでたっても終了しないこともある。それでも計算が終了するときにはその手続は f の正しい値を出力してくれる。」この直感に相当するのが**部分再帰的関数**の概念である。

部分再帰的関数は部分関数の概念に基づいて定義される。**部分関数**とは、大雑把にいって全てのインプットについてアウトプットが存在するとは限らないような “関数” のことである。(正確な定義は 2 章を参照。) 例えば割り算 / は自然数上の演算として見た場合、部分関数である。なぜならば n が m で割り切れない場合には n/m は定義されていないからである。同様に / は有理数上の演算としても部分関数である。なぜならば分母が 0 の場合 $n/0$ が定義されていないからである。一方、/ は 0 以外の有理数の集合上では関数であるといえる。

まず、予備的な定義からはじめる。

定義 4.3

- f を部分関数とし、$\vec{n} \in \mathbb{N}$ とするとき、$f(\vec{n})$ の値が定義されているときには $f(\vec{n})\downarrow$ と書き、さもなければ $f(\vec{n})\uparrow$ と書く。
- $f(\vec{x})$、$g(\vec{x})$ を部分関数とするとき、$f(\vec{x}) \sim g(\vec{x})$ が成り立つのは、どんな $\vec{n} \in \mathbb{N}$ についても

  1. $f(\vec{n})\downarrow \Longleftrightarrow g(\vec{n})\downarrow$
  2. $f(\vec{n})\downarrow$ ならば $f(\vec{n}) = g(\vec{n})$

  が成り立つときである。

部分関数 f が**全域的** (total) といわれるのは、どんな $\vec{n} \in \mathbb{N}$ についても $f(\vec{n})\downarrow$ が成り立つ場合である。全域的な部分関数とは、すなわち関数のことに他ならない。

部分再帰的関数とは、大雑把に言って再帰的関数の定義 (4.1) から実効性条件を取り除くことにより得られるもののことである。

定義 4.4 部分再帰的関数 *(partially recursive functions)* の集合は次のように帰納的に定義される。

1. ゼロ関数、射影関数、後続者関数は部分再帰的である。
2. **合成** *(composition)* : g が m 項部分再帰的関数であり、$h_1, \ldots, h_m$ が n 項部分再帰的関数ならば、

$$f(x_1, \ldots, x_n) \sim g(h_1(x_1, \ldots, x_n), \ldots, h_m(x_1, \ldots, x_n))$$

   により定義される関数 f は部分再帰的関数である。ただし $g(h_1(x_1, \ldots, x_n), \ldots, h_m(x_1, \ldots, x_n))$ は $h_1(\vec{x}) = z_1, \ \ldots, \ h_m(\vec{x}) = z_m$ かつ $g(z_1, \ldots, z_m)$ が定義されているときにのみ定義されており、値 z をとるものとする。
3. **原始再帰法** *(primitive recursion)* : g が n 項部分再帰的関数であり、h が $n+2$ 項部分再帰的関数ならば、

$$\begin{aligned} f(x_1, \ldots, x_n, 0) &\sim g(x_1, \ldots, x_n) \\ f(x_1, \ldots, x_n, y+1) &\sim h(x_1, \ldots, x_n, y, f(x_1, \ldots, x_n, y)) \end{aligned}$$

   により定義される関数 f も部分再帰的である。ただし $h(x_1, \ldots, x_n, y, f(x_1, \ldots, x_n, y))$ は $f(\vec{x}, y) = z_0$ かつ $h(\vec{x}, y, z_0) = z$ が定義されているときにのみ定義されており、値 z をとるものとする。（このことからすぐわかるように、ある n について $f(\vec{x}, n) \uparrow$ ならば、全ての $m \geq n$ について $f(\vec{x}, m) \uparrow$ である。）
4. **最小化**（*minimization*）: g が $n+1$ 項部分再帰的関数のとき

$$f(x_1, \ldots, x_n) \sim \mu y(g(x_1, \ldots, x_n, y) = 0)$$

   により定義される関数 f は部分再帰的である。ここで $\mu y(g(x_1, \ldots, x_n, y) = 0)$ は $g(x_1, \ldots, x_n, y) = 0$ が成り立つような y が存在するときにはその中で最小のものを表し、さもなければ値が定義されていないものとする。（ここでは実効性条件は仮定されていないことに注意。この点が再帰的関数との最大の違いである。）

明らかに全ての再帰的関数は部分再帰的関数でもある。再帰的関数の場合と同様に、次の定理が成り立つ。

定理 4.5 次のように定義される部分再帰的’関数のクラスは部分再帰的関数のクラスと一致する。

1. ゼロ関数、射影関数、後続者関数、$+, \cdot$ および $\chi_=$ は部分再帰的’である。
2. 部分再帰的’関数から合成および（実効性条件なしの）最小化により得られる関数は再帰的’である。

ゆえに以後部分再帰的関数と部分再帰的 ’ 関数を同一視してよい。

4.3 標準形定理

3.4 節では順序対、有限集合、有限列などを自然数へとコード化したが、ここではさらに部分再帰的関数を自然数へとコード化する方法について述べる。一般に部分再帰的関数のコードは指標と呼ばれているので、ここでも指標という呼び方を用いる。さて、(部分) 関数 f を自然数 e でコード化するということが意味をなすためには、e が与えられたとき、そこから f の入出力関係を正確に復元できるのでなければならない。そのためには、指標 e として選ばれる数はどんな自然数でもよいわけではなく、それは関数の計算の仕方（インプットが与えられたときアウトプットを求めるにはどうしたらよいか)、あるいは別の言い方をすれば関数の設計図（すなわちその関数がどのように構成されているか）を情報として含んでいるのでなければならない。つまり、関数の指標は関数を計算するためのプログラムとみなすことができるのでなければならない。

本章の主定理はクリーネの標準形定理と呼ばれるものであるが、その核心はまさに、指標が与えられたときにそこから関数を復元する方法を（一群の）部分再帰的関数により与えるという点にある。すなわち、どんな関数 f についてもその指標 e さえ与えられれば、f の入出力関係を正確にシミュレートできる、そういう**万能な**部分再帰的関数を構成する。今、指標 e のことを関数 f を計算するためのプログラムだと思うことにすれば、ここで構成する万能な部分関数はプログラムを解釈し実行するインタプリタに相当するものと考えることができる。

このようなコード化を行うためには有限列のコードを用いることが必要である。我々はすでに二つのコード化の手法（すなわち 3.2 節の素因数分解の一意性を用いた方法と 3.4 の部分列・区切り数などに依拠した方法）を導入したが、ここでは前者を用いることにする。(後者を用いても同等のことができる。) 3.2 節では以下の原始再帰的関数・関係を定義したことを思い出しておこう。

1. 自然数の列 $n_1, \ldots, n_k$ が与えられたとき、別の自然数 $\langle n_1, \ldots, n_k \rangle$ を割り当てる関数。
2. 列のコード x が与えられたとき、その i 番目の要素を取り出す 2 項関数 $(x)_i$. とくに $1 \leq i \leq k$ について
$$(\langle n_1, \ldots, n_k \rangle)_i = n_i.$$
3. コード x の長さを返す関数 $\mathsf{leng}(x)$。とくに次のことが成り立つ:
$$\mathsf{leng}(\langle n_1, \ldots, n_k \rangle) = k.$$
4. 「x が列のコードである」ことを表す関係 $\mathsf{seq}(x)$. とくに $\mathsf{seq}(\langle n_1, \ldots, n_k \rangle)$ は真である。

定理 4.5 により、どんな部分再帰的関数も初期関数 zero, suc, proj, $+$, $\cdot$, $\chi_=$ から始めて合成や最小化を繰り返し用いることにより定義できる。これを自然数であらわすには、まず初期関数に指標（自然数）を割り当て、次に単純な関数の指標からより複雑な関数の指標を構成するための規則を述べればよい。

定義 4.6 部分再帰的'関数 f の**指標** *(index)* とは、次のように定義される自然数のことである。

1. $\lceil zero^n \rceil = \langle 0, n \rangle$ は zero^n の指標である。
2. $\lceil suc \rceil = \langle 1, 1 \rangle$ は suc の指標である。
3. $\lceil proj_i^n \rceil = \langle 2, n, i \rangle$ は proj_i^n の指標である。
4. $\lceil plus \rceil = \langle 3, 2 \rangle$ は $+$ の指標である。
5. $\lceil mult \rceil = \langle 4, 2 \rangle$ は $\cdot$ の指標である。
6. $\lceil equal \rceil = \langle 5, 2 \rangle$ は $\chi_=$ の指標である。
7. f が部分再帰的関数 $g, h_1, \ldots, h_m$ から合成により定義されているとする。すなわち $f(\vec{x}) = g(h_1(\vec{x}), \ldots, h_m(\vec{x}))$。$\lceil g \rceil, \lceil h_1 \rceil, \ldots, \lceil h_m \rceil$ がそれぞれ $g, h_1, \ldots, h_m$ の指標のとき、
   $\lceil comp(g, h_1, \ldots, h_m) \rceil = \langle 6, n, \lceil g \rceil, \lceil h_1 \rceil, \ldots, \lceil h_m \rceil \rangle$ は f の指標である（ここで n は f の項数)。
8. f が部分再帰的関数 g から最小化により定義されているとする。このとき、$\lceil min(g) \rceil = \langle 7, n, \lceil g \rceil \rangle$ は f の指標である（ここで n は f の項数)。

部分再帰的関数の指標は唯一つには定まらない。なぜならば一つの関数を構成するのにはさまざまな方法があり、それぞれに対して別々の指標が割り当てられるからである。また、どんな部分再帰的関数の指標でもない自然数が存在する。それには三つの場合がある。第一に自然数 n がどのような有限列をも表さない場合、すなわち $\neg\mathsf{seq}(n)$ の場合であり、第二に n は何らかの有限列を表すが、上の形に当てはまらない場合である。

一項関数 arity と一項関係 wf を次のように定義する。

$$
\begin{aligned}
\mathsf{arity}(x) &= (x)_2 \\
\mathsf{wf}(x) &\iff \mathsf{seq}(x) \wedge (0 \leq (x)_1 \leq 7) \\
&\quad \wedge((x)_1 = 0 \to \mathsf{leng}(x) = 2) \\
&\quad \wedge((x)_1 = 1 \to \mathsf{leng}(x) = 2 \wedge \mathsf{arity}(x) = 1) \\
&\quad \wedge((x)_1 = 2 \to \mathsf{leng}(x) = 3 \wedge (x)_3 \leq \mathsf{arity}(x)) \\
&\quad \wedge(3 \leq (x)_1 \leq 5 \to \mathsf{leng}(x) = 2 \wedge \mathsf{arity}(x) = 2) \\
&\quad \wedge((x)_1 = 6 \to \mathsf{leng}(x) = \mathsf{arity}((x)_2) + 3 \wedge \forall 1 \leq y \leq m(\mathsf{arity}((x)_{y+2}) = \mathsf{arity}(x))) \\
&\quad \wedge((x)_1 = 7 \to \mathsf{leng}(x) = 3 \wedge \mathsf{arity}(x) + 1 = \mathsf{arity}((x)_3)
\end{aligned}
$$

直感的にいえば、$\mathsf{wf}(e)$ は「自然数 e は一番外側だけを見れば指標の定義の形に合致している」ことを述べている。特に e を関数 f の指標とすると、$\mathsf{wf}(e)$ が成り立つ（しかし $\mathsf{wf}(e)$ が成り立つからといって、必ずしも e が指標であるとは限らない)。$\mathsf{arity}(e)$ は e が関数 f の指標のときには、関数 f の項数を表す。

$\lceil f \rceil$ を k 項関数の指標とし、$\vec{m} = m_1, \ldots, m_k$、$n$ を自然数とするとき、自然数 $\langle \lceil f \rceil, \vec{m}, n \rangle$ のことを $\lceil f(\vec{m}) = n \rceil$ と書いて表す。表記法から推察できるように、この自然数は等式 $f(\vec{m}) = n$ を表すコードである。

次に 1 項関数 fname, output、1 項関係 equation を次のように定義する。

$$\begin{aligned}
\mathsf{fname}(x) &= (x)_1 \\
\mathsf{output}(x) &= (x)_{\mathsf{leng}(x)} \\
\mathsf{equation}(x) &\iff \mathsf{wf}(\mathsf{fname}(x)) \wedge \mathsf{leng}(x) = \mathsf{arity}(\mathsf{fname}(x)) + 2
\end{aligned}$$

すると $\mathsf{equation}(\ulcorner f(\vec{m}) = n \urcorner)$ は常に成り立ち、また

$$\begin{aligned}
\mathsf{fname}(\ulcorner f(\vec{m}) = n \urcorner) &= \ulcorner f \urcorner \\
\mathsf{output}(\ulcorner f(\vec{m}) = n \urcorner) &= n
\end{aligned}$$

が成り立つ。

$\ulcorner comp(g, h_1, \ldots, h_k)(\vec{m}) = n \urcorner$ の形の自然数が与えられたとき、次の自然数をこの等式コードの**証拠**と呼ぶことにする。

$$\langle \ulcorner h_1(\vec{m}) = n_1 \urcorner, \ldots, \ulcorner h_k(\vec{m}) = n_k \urcorner, \ulcorner g(n_1, \ldots, n_k) = n \urcorner \rangle$$

もしもこのような自然数が存在し、かつ $h_1(\vec{m}) = n_1$, …, $h_k(\vec{m}) = n_k$, $g(n_1, \ldots, n_k) = n$ がすべて成り立つならば、$\ulcorner comp(g, h_1, \ldots, h_m) \urcorner$ が表す関数 f について $f(\vec{m}) = n$ が成り立つ。ゆえにこの自然数は $f(\vec{m}) = n$ が成り立つためのまさに “証拠” なのである。

$\ulcorner min(g)(\vec{m}) = n \urcorner$ の形の自然数が与えられたとき、次の自然数をこの等式コードの**証拠**と呼ぶ。

$$\langle \ulcorner g(\vec{m}, 0) = k_0 \urcorner, \ldots, \ulcorner g(\vec{m}, n-1) = k_{n-1} \urcorner, \ulcorner g(\vec{m}, n) = 0 \urcorner \rangle$$

ただし $k_0, \ldots, k_{n-1} > 0$ とする。上と同じ意味で、この自然数は $\mu z(g(\vec{m}, z) = 0) = n$ が成り立つことの “証拠” になっている。

二項関係 cwitness と muwitness を次のように定義する。

$$\begin{aligned}
\mathsf{cwitness}(x, y) \iff\ & \mathsf{seq}(y) \wedge (\forall 1 \le z \le \mathsf{leng}(y).\mathsf{equation}((y)_z)) \\
& \wedge \forall 1 \le z \le \mathsf{leng}(y) \dot{-} 1((\mathsf{fname}(x))_{z+3} = \mathsf{fname}((y)_z) \\
& \quad \wedge \forall 1 \le w \le \mathsf{arity}(\mathsf{fname}(x))((x)_{w+1} = ((y)_z)_{w+1}) \\
& \quad \wedge \mathsf{output}((y)_z) = ((y)_{\mathsf{leng}(y)})_z) \\
& \wedge (\mathsf{fname}(x))_3 = \mathsf{fname}((y)_{\mathsf{leng}(y)}) \wedge (\mathsf{output}((y)_{\mathsf{leng}(y)}) = \mathsf{output}(x)). \\
\mathsf{muwitness}(x, y) \iff\ & \mathsf{seq}(y) \wedge (\forall 1 \le z \le \mathsf{leng}(y).\mathsf{equation}((y)_z) \wedge (\mathsf{fname}(x))_3 = \mathsf{fname}((y)_z) \\
& \wedge \forall 1 \le w \le \mathsf{arity}(\mathsf{fname}(x))((x)_{w+1} = ((y)_z)_{w+1}) \\
& \wedge (((y)_z)_{\mathsf{arity}(\mathsf{fname}(x))+1} + 1 = z) \wedge (z \ne \mathsf{leng}(y) \to \mathsf{output}((y)_z) > 0)) \\
& \wedge (\mathsf{leng}(y) = \mathsf{output}(x) + 1).
\end{aligned}$$

このように定義すると次のことが成り立つ: $eq_1 = \ulcorner comp(g, h_1, \ldots, h_k)(\vec{m}) = n \urcorner$, $eq_2 = \ulcorner min(g)(\vec{m}) = n \urcorner$ のとき、

$$\begin{aligned}
\mathsf{cwitness}(eq_1, w) &\iff w \text{ が } eq_1 \text{ の証拠である。} \\
\mathsf{muwitness}(eq_2, w) &\iff w \text{ が } eq_2 \text{ の証拠である。}
\end{aligned}$$

自然数 $\langle q_1, \ldots, q_r \rangle$ が q_r の**計算列**と呼ばれるのは、各 q_i が次の条件を満たす場合である：

1. q_i は等式のコードである。
2. $\mathsf{fname}(q_i) = \lceil zero^k \rceil$ ならば $q_i = \lceil zero^k(\vec{m}) = 0 \rceil$ の形である。
3. $\mathsf{fname}(q_i) = \lceil suc \rceil$ ならば $q_i = \lceil suc(m) = m + 1 \rceil$ の形である。
4. $\mathsf{fname}(q_i) = \lceil proj^k_j \rceil$ ならば $q_i = \lceil proj^k_j(m_1, \ldots, m_k) = m_j \rceil$ の形である。
5. $\mathsf{fname}(q_i) = \lceil plus \rceil$ ならば $q_i = \lceil plus(m_1, m_2) = m_1 + m_2 \rceil$ の形である。
6. $\mathsf{fname}(q_i) = \lceil mult \rceil$ ならば $q_i = \lceil mult(m_1, m_2) = m_1 \cdot m_2 \rceil$ の形である。
7. $\mathsf{fname}(q_i) = \lceil equal \rceil$ ならば $q_i = \lceil equal(m_1, m_2) = \chi_=(m_1, m_2) \rceil$ の形である。
8. $\mathsf{fname}(q_i) = \lceil comp(g, h_1, \ldots, h_k) \rceil$ または $\mathsf{fname}(q_i) = \lceil min(g) \rceil$ ならば、q_i の証拠はすべて $q_1, \ldots, q_{i-1}$ の中に現れる。

補題 4.7 $\lceil f \rceil$ を部分再帰的関数 f の指標とする。このとき、

$f(\vec{m}) = n$ である $\Longleftrightarrow \lceil f(\vec{m}) = n \rceil$ の計算列が存在する。

証明 ($\Rightarrow$) 部分再帰的' 関数 f の構成に関する帰納法による。
($\Leftarrow$) k を何らかの等式 $\lceil g(\vec{m}') = n' \rceil$ の計算列とするとき、$g(\vec{m}') = n'$ となることを $\mathsf{leng}(k)$ に関する帰納法により証明すればよい。ここで $\lceil g \rceil$ は g の指標である。 ■

1 項関係 $\mathsf{compseq}(x)$、1 項関数 $\mathsf{main}(x)$ を次のように定義する。

$$
\begin{aligned}
\mathsf{compseq}(x) \iff\ & \mathsf{seq}(x) \wedge \forall 1 \le i \le \mathsf{leng}(x)(\mathsf{equation}(x) \\
& \wedge \forall k < x(\mathsf{fname}((x)_i) = \lceil zero^k \rceil \to \mathsf{output}((x)_i) = 0) \\
& \wedge (\mathsf{fname}((x)_i) = \lceil suc \rceil \to \mathsf{output}((x)_i) = ((x)_i)_2 + 1) \\
& \wedge \forall k_1 < x \forall k_2 < x(\mathsf{fname}((x)_i) = \lceil proj^{k_1}_{k_2} \rceil \to \mathsf{output}((x)_i) = ((x)_i)_{k_2+1}) \\
& \wedge (\mathsf{fname}((x)_i) = \lceil plus \rceil \to \mathsf{output}((x)_i) = ((x)_i)_2 + ((x)_i)_3) \\
& \wedge (\mathsf{fname}((x)_i) = \lceil mult \rceil \to \mathsf{output}((x)_i) = ((x)_i)_2 \cdot ((x)_i)_3) \\
& \wedge (\mathsf{fname}((x)_i) = \lceil equal \rceil \to \mathsf{output}((x)_i) = \chi_=(((x)_i)_2, ((x)_i)_3)) \\
& \wedge ((\mathsf{fname}((x)_i))_1 = 6 \to \exists y < x(\mathsf{cwitness}(y, (x)_i) \\
& \quad \wedge \forall 1 \le k \le \mathsf{leng}(y) \exists 1 \le j < i((y)_k = (x)_j))) \\
& \wedge ((\mathsf{fname}((x)_i))_1 = 7 \to \exists y < x(\mathsf{muwitness}(y, (x)_i) \\
& \quad \wedge \forall 1 \le k \le \mathsf{leng}(y) \exists 1 \le j < i((y)_k = (x)_j))).
\end{aligned}
$$

$$
\mathsf{main}(x) \quad = \quad (x)_{\mathsf{leng}(x)}.
$$

このように定義すると、次のことが成り立つ：eq を等式のコードとするとき、

$$\mathsf{compseq}(w) \wedge \mathsf{main}(w) = eq \Longleftrightarrow w \text{ は } eq \text{ の計算列である。}$$

次の定理はクリーネの**標準形定理** (normal form theorem) と呼ばれている。これは再帰的関数論の基本定理であるといってよい。

定理 4.8 次のような性質を持つ *1* 項原始再帰的関数 U と原始再帰的関係 $\mathsf{T}_1, \mathsf{T}_2, \ldots$ が存在する（ここで各 T_i は $i+2$ 項関係)。e を i 項部分再帰的関数 f の指標とするとき、

$$f(\vec{x}) \sim \mathsf{U}(\mu y \mathsf{T}_i(e, \vec{x}, y))$$

ここで U や各 T_i は原始再帰的であるから、次のことが帰結する：

- どんな部分再帰的関数も、最小化を高々1 回使えば定義することができる。

証明 T_i, U を次のように定義すればよい。

$$\begin{aligned} \mathsf{T}_i(e, \vec{x}, y) &\iff \mathsf{compseq}(y) \wedge \exists z < y(\mathsf{main}(y) = \langle e, \vec{x}, z \rangle). \\ \mathsf{U}(y) &= \mathsf{output}(\mathsf{main}(y)). \end{aligned}$$

さて、f を i 項部分再帰的関数とし、e をその指標とする。もしも $f(\vec{m}) = n$ ならば、補題 4.7 により $\lceil e(\vec{m}) = n \rceil$ の計算列が存在する。その中でも最小の計算列を w とする。このとき $\mathsf{compseq}(w) \wedge \mathsf{main}(w) = \langle e, \vec{m}, n \rangle$ であり、当然 $n < w$ であるから、$\mathsf{T}_i(e, \vec{m}, w)$ が成り立つ。w の最小性により、

$$\mathsf{U}(\mu y \mathsf{T}_i(e, \vec{m}, y)) = \mathsf{U}(w) = \mathsf{output}(\langle e, \vec{m}, n \rangle) = n.$$

これで $f(\vec{m}) = n$ ならば $\mathsf{U}(\mu y \mathsf{T}_i(e, \vec{m}, y)) = n$ となることが示された。逆方向も同様に証明できる。 ■

練習問題 4.9

1. *(**)* 定義 *4.6* では再帰的'関数の構成法にのっとって指標を定義した。代わりに（原始再帰法を含む）もともとの再帰的関数の構成法にのとって指標を定義し、標準形定理を証明せよ。

2. *(*)* 上の問題に基づいて、「x は原始再帰的関数の指標である」という性質は原始再帰的であることを証明せよ。（この事実は、後で証明する次の事実と対比すると面白い：「x は再帰的関数の指標である」という性質は再帰的ではない。）

3. *(*)* 補題 *4.7* を証明せよ。

4.4 標準形定理の帰結

（全域）再帰的関数を構成するときには、最小化を用いるたびに逐一実効性条件を判定する必要があった。次の定理が示しているのは、このように実効性をいちいち判定せずとも、全体として得られる部分再帰的関数が全域的でさえあれば、それは再帰的関数であるといってよいということである。

定理 4.10 全域的な部分再帰的関数は再帰的関数である。

証明　f を i 項の全域的な部分再帰的関数とする。e を f の指標とすると、定理 4.8 により、$f(\vec{x}) \sim \mathsf{U}(\mu y \mathsf{T}_i(e, \vec{x}, y))$ が成り立つ。f は全域的であるから、$\mu y \mathsf{T}_i(e, \vec{x}, y)$ はどんな $\vec{x}$ についても定義されていなければならず、ゆえに $\mathsf{T}_i(e, \vec{x}, y)$ は実効性条件を満たしているのでなければならない。ゆえに $\mathsf{U}(\mu y \mathsf{T}_i(e, \vec{x}, y))$ は再帰的関数である。 ■

上の定理があるので、再帰的関数とは部分再帰的関数の中で全域的であるもののことに過ぎないと考えることができる。ゆえに今後は部分再帰的関数一般について話を進めることにする。

3.3 節では、原始再帰的関数の限界について考察した。特に原始再帰的関数の列 Υ_0, Υ_1, ... を構成し、そこから定義できる関数 $\Upsilon(x) = \Upsilon_x(x)$ は原始再帰的ではないことを証明した。一方、標準形定理を用いれば次のことが証明できる。

命題 4.11　Υ は再帰的関数である。

証明　各 $n \geq 0$ について Υ_n は原始再帰的関数であるから、指標 $\mathsf{ups}(n)$ を持つ。しかも $\mathsf{ups}(n)$ を n についての 1 項関数とみたとき、ups が原始再帰的であることも証明することができる。

標準形定理により、

$$\Upsilon_w(x) \sim \mathsf{U}(\mu y \mathsf{T}_1(\mathsf{ups}(w), x, y))$$

よって $\Upsilon_w(x)$ は部分再帰的関数である。しかも全域的であるから、定理 4.10 により再帰的関数であることがわかる。よって $\Upsilon(x) = \Upsilon_x(x)$ も再帰的関数である。 ■

Υ は原始再帰的ではないが再帰的である。ゆえに再帰的関数のクラスは原始再帰的関数のクラスよりも真に大きいことがわかる。

練習問題 4.12　練習問題 *3.28* に出てきたアッカーマン関数が再帰的関数であることを証明せよ。

指標、つまり何らかの部分再帰的関数のコードとなっている数の集合を考える。各指標は自然数であるから、自然数の大小関係に従って、指標は一列に並べることができる。いま、n 番目の部分再帰的関数の指標を $\mathsf{parrecindex}(n)$ と書くことにすれば、$\mathsf{parrecindex}(x)$ は x についての 1 項関数としてみることができる。すると次のことが成り立つ。

命題 4.13 関数 $\mathsf{parrecindex}(x)$ は原始再帰的である。また、x が *1* 項部分再帰的関数の指標であるときに限って成立する性質 $\mathsf{parrecindex?}(x)$ も原始再帰的である。

練習問題 4.14 *(*)* 命題 *4.13* を証明せよ。

しかし同様のことは「全域」再帰的関数の指標の集合については成り立たない。上と同様に n 番目の再帰的関数の指標を $\mathsf{recindex}(n)$ と書くことにすると、標準形定理とカントールの対角線論法を組み合わせることにより、次のことが証明できる。

定理 4.15 関数 $\mathsf{recindex}(x)$ は再帰的ではない。

証明 仮に $\mathsf{recindex}(x)$ が再帰的関数だとして、2 項関数 $\mathsf{value}(w,x)$ を次のように定義する :

$$\mathsf{value}(w,x) = \mathsf{U}(\mu y \mathsf{T}_1(\mathsf{recindex}(w), x, y)).$$

$\mathsf{value}(w,x)$ は、「w 番目の 1 項再帰的関数にインプット x を与えたときのアウトプット」を表す。$\mathsf{recindex}(w)$ は再帰的であり、なおかつどんな w に対しても常に何らかの再帰的関数の指標を返すので、実効性条件 $\forall w \forall x \exists y \mathsf{T}_1(\mathsf{recindex}(w), x, y))$ が成り立つ。ゆえに $\mathsf{value}(w,x)$ は再帰的であることになる。

ゆえに次のように対角化により定義される 1 項関数 $\overline{\mathsf{value}}$ も再帰的である :

$$\overline{\mathsf{value}}(x) = \mathsf{value}(x,x) + 1.$$

いま、$\overline{\mathsf{value}}$ は k 番目の 1 項再帰的関数であるとすると、定義により $\mathsf{value}(k,x) = \overline{\mathsf{value}}(x)$ が成り立つはずである。しかしこれは不合理。なぜならば、

$$\overline{\mathsf{value}}(k) = \mathsf{value}(k,k) + 1 = \overline{\mathsf{value}}(k) + 1$$

となり矛盾するからである。ゆえに $\mathsf{recindex}$ は再帰的ではない。 ■

ここで 2 項関数 value から 1 項関数 $\overline{\mathsf{value}}$ を構成する方法が対角化と呼ばれるのは次の理由による。いま $\mathsf{value}(n,m)$ の値（$n = 0, 1, \ldots; m = 0, 1, \ldots$）を表

	0	1	$\cdots$	m	$\cdots$
0	$\mathsf{value}(0,0)$	$\mathsf{value}(0,1)$	$\cdots$	$\mathsf{value}(0,m)$	$\cdots$
1	$\mathsf{value}(1,0)$	$\mathsf{value}(1,1)$	$\cdots$	$\mathsf{value}(1,m)$	$\cdots$
$\vdots$	$\vdots$	$\vdots$		$\vdots$	
n	$\mathsf{value}(n,0)$	$\mathsf{value}(n,1)$	$\cdots$	$\mathsf{value}(n,m)$	$\cdots$
$\vdots$	$\vdots$	$\vdots$		$\vdots$	

で表したとき、$\overline{\mathsf{value}}(n)$ は対角線上の n 番目の値 $\mathsf{value}(n,n)$ に 1 を加えたものに相当するからである。

系 4.16 次のように定義される性質 recindex? は再帰的ではない：

$$\mathsf{recindex?}(n) \Longleftrightarrow n \text{ は } \textit{1} \text{ 項再帰的関数の指標である。}$$

証明　性質 recindex? を用いれば、1 項関数 recindex は次のように定義できる：

$$\begin{aligned} \mathsf{recindex}(0) &= \mu x.\mathsf{recindex?}(x) \\ \mathsf{recindex}(y+1) &= \mu x(\mathsf{recindex}(y) < x \wedge \mathsf{recindex?}(x)). \end{aligned}$$

ここで $\mathsf{recindex}(y) < x \wedge \mathsf{recindex?}(x)$ に関して実効性条件が成り立つこと、すなわちどんな y が与えられても、この性質を満たす x が常に存在することは明らかである。ゆえにもしも recindex? が再帰的だとすると、recindex も再帰的になり、前の定理に反する。 ■

性質 parrecindex? は原始再帰的であるが、recindex? は再帰的でない。両者を分けているのは、最小化を用いる際に実効性を判定するかどうかの違いであり、この点が困難の核心であると言える。

4.5　再帰定理

$n+1$ 項部分再帰的関数 $\{e\}^n(x_1,\ldots,x_n)$ を

$$\{e\}^n(\vec{x}) \sim \mathsf{U}(\mu y \mathsf{T}_n(e,\vec{x},y))$$

により定義する。ここで U, T_n は定理 4.8 に現れたものである。すなわち、$\{e\}^n(x_1,\ldots,x_n)$ は、もしも e が何らかの n 項部分再帰的関数 f の指標の場合には $f(x_1,\ldots,x_n)$ と一致し、そうでなければ（未定義の場合も含めて）任意の値をとるような部分関数である。

次の定理は**パラメータ定理** (parameter theorem) または **s-m-n 定理**と呼ばれる。

定理 4.17 任意の n について原始再帰的関数 S_n が存在し、

$$\{e\}^{n+1}(x_1,\ldots,x_n,y) \sim \{\mathsf{S}_n(e,y)\}(x_1,\ldots,x_n)$$

が成り立つ。より一般的に、任意の $m,n \geq 0$ について原始再帰的関数 S_n^m が存在し、

$$\{e\}^{m+n}(x_1,\ldots,x_n,y_1,\ldots,y_m) \sim \{\mathsf{S}_n^m(e,y_1,\ldots,y_m)\}(x_1,\ldots,x_n)$$

が成り立つ。

証明　ここでは一番目の主張のみを示す。

定数関数 $\mathsf{const}_k^n(\vec{x}) = k$ は原始再帰的関数である。この関数の指標を $\lceil const_k^n \rceil$ と書くことにすると、n を固定したとき $\lceil const_y^n \rceil$ は y についての 1 項関数としてみることができる。しかもこの関数は原始再帰的であることが証明できる。S_n を次のように定義する:

$$\mathsf{S}_n(e,y) = \langle 6,n,e,\lceil proj_1^n \rceil,\ldots,\lceil proj_n^n \rceil,\lceil const_y^n \rceil \rangle.$$

すると、自然数 e, p が与えられたとき、もしも e が $n+1$ 項部分関数 $f(\vec{x}, y)$ の指標ならば $\mathsf{S}_n(e, p)$ は n 項部分再帰的関数 $f(\vec{x}, \mathsf{const}_p^n(\vec{x}))$ の指標となっていることがわかる。ゆえに

$$\begin{aligned} \{e\}^{n+1}(\vec{x}, p) = q &\iff f(\vec{x}, p) = q \\ &\iff f(\vec{x}, \mathsf{const}_p^n(\vec{x})) = q \\ &\iff \{\mathsf{S}_n(e, p)\}^n(\vec{x}) = q. \end{aligned}$$

（より厳密には、e がたとえ指標でなくても、（何らかの事故的な理由で）$\{e\}^{n+1}(\vec{x}, p) = q$ となる場合には $\{\mathsf{S}_n(e, p)\}^n(\vec{x}) = q$ となり、逆もまた真になるということを示さねばならないが、ここでは割愛する。） ■

この定理の帰結として次の**再帰定理** (recursion theorem)、あるいは**不動点定理** (fixpoint theorem) が得られる。

定理 4.18 $f(x_1, \ldots, x_n, y)$ を部分再帰的関数とするとき、ある自然数 fix_f が存在して、

$$\{\mathsf{fix}_f\}^n(x_1, \ldots, x_n) \sim f(x_1, \ldots, x_n, \mathsf{fix}_f)$$

が成り立つ。

証明 $\mathsf{omega}(y) = \mathsf{S}_n(y, y)$ と定義する。すると定理 4.17 によりどんな y についても

$$\{y\}^{n+1}(\vec{x}, y) \sim \{\mathsf{S}_n(y, y)\}^n(\vec{x}) \sim \{\mathsf{omega}(y)\}^n(\vec{x})$$

が成り立つ。$f(\vec{x}, \mathsf{omega}(y))$ は部分再帰的関数であるから、指標 e が存在する。このとき、$\mathsf{fix}_f = \mathsf{omega}(e)$ と定義する。すると、

$$\{\mathsf{fix}_f\}^n(\vec{x}) \sim \{\mathsf{omega}(e)\}^n(\vec{x}) \sim \{e\}^{n+1}(\vec{x}, e) \sim f(\vec{x}, \mathsf{omega}(e)) \sim f(\vec{x}, \mathsf{fix}_f)$$

が成り立つ。 ■

4.6 再帰的枚挙可能関係

再帰的関数を特性関数とするような関係のことを再帰的関係と呼んだ。すなわち再帰的関数からはじめて、それに対応する関係を考え、それを再帰的関係と呼んだわけである。同じような考え方により、部分再帰的関数に対応する関係の概念を定義することができる。

定義 4.19 n 項関係 $R \subseteq \mathbb{N}^n$ が**再帰的枚挙可能** *(recursively enumerable)* であるのは、次のように定義される n 項部分関数 χ'_R が部分再帰的関数の場合である：

$$\begin{aligned} \chi'_R(\vec{x}) &= 1 && R(\vec{x}) \text{ が成り立つとき} \\ &= \text{未定義} && \text{そうでないとき} \end{aligned}$$

命題 4.20 再帰的関係は全て再帰的枚挙可能関係である。

証明　$R \subseteq \mathbb{N}^n$ が再帰的関係であるとする。このとき、χ'_R は次のように定義することができる。

$$\chi'_R(\vec{x}) = \mu z(R(\vec{x}) \wedge z = 1).$$

■

「再帰的枚挙可能」の名前は次の性質に由来する。

命題 4.21　$R \subseteq \mathbb{N}$ を空でない集合とする。このとき R が再帰的枚挙可能集合であるのは次のような原始再帰的関数 enum_R が存在する場合である：

$$R(n) \Longleftrightarrow \text{ある } m \in \mathbb{N} \text{ について } \mathsf{enum}_R(m) = n$$

すなわち $R = \{\mathsf{enum}_R(m) | m \in \mathbb{N}\}$ が成り立つ場合である。

より一般的に、$R \subseteq \mathbb{N}^n$ を空でない n 項関係とする。このとき R が再帰的枚挙可能であるのは次のような原始再帰的関数 enum_R が存在する場合である：

$$R(\vec{n}) \Longleftrightarrow \text{ある } m \in \mathbb{N} \text{ について } \mathsf{enum}_R(m) = \langle \vec{n} \rangle.$$

証明　最初の主張のみ証明する。$R \subseteq \mathbb{N}$ を空でない再帰的枚挙可能集合とすると、部分再帰的関数 χ'_R が存在して、

$$\begin{array}{rcll} \chi'_R(x) & = & 1 & R(x) \text{ が成り立つとき} \\ & = & \text{未定義} & \text{そうでないとき} \end{array}$$

となる。e を χ'_R の指標とすれば、標準形定理（定理 4.8）により原始再帰的関数 U と T が存在して

$$\chi'_R(x) \sim \mathsf{U}(\mu y \mathsf{T}(e, x, y))$$

が成り立つ。ゆえに任意の $n \in \mathbb{N}$ について、

$$(*) \quad R(n) \Longleftrightarrow \text{ある } k \in \mathbb{N} \text{ について } \mathsf{T}(e, n, k)$$

である。

仮定により R は空ではないので、$R(d)$ となる要素 d が存在する。このとき、関数 enum_R を次のように定義する。

$$\begin{array}{rcll} \mathsf{enum}_R(z) & = & \pi_1(z) & \mathsf{T}(\pi_1(z), \pi_2(z)) \text{ が成り立つとき} \\ & = & d & \text{そうでないとき} \end{array}$$

ここで π_1, π_2 は 3.4 節で定義された、順序対をコード化する関数 $\langle x_1, x_2 \rangle$ の逆関数である。すなわち、$\pi_i(\langle x_1, x_2 \rangle) = x_i$ $(i = 1, 2)$。このように定義された enum_R が原始再帰的関数であることは明らかなので、あとはこの関数が求められる性質を満たすことを示せばよい。

まず、自然数 n について $R(n)$ が成り立つとする。すると、$(*)$ により、ある $k \in \mathbb{N}$ について $\mathsf{T}(e, n, k)$ である。$m = \langle n, k \rangle$ とすれば、$\mathsf{T}(\pi_1(m), \pi_2(m))$ が成り立つ。すなわち、$\mathsf{enum}_R(m) = \pi_1(m) = n$ である。

逆に、ある m について $\mathsf{enum}_R(m) = n$ であるとする。もしも $\mathsf{T}(\pi_1(m), \pi_2(m))$ が成り立つならば、定義により $n = \pi_1(m)$ であり、(*) により、$R(n)$ となる。また $\mathsf{T}(\pi_1(m), \pi_2(m))$ が成り立たないときには、$n = d$ で $R(d)$ であるから、やはり $R(n)$ となる。 ■

このことが意味するのは、R の要素は、原始再帰的関数 enum_R によって、「0 番目の要素」$\mathsf{enum}_R(0)$, 「1 番目の要素」$\mathsf{enum}_R(1)$, 「2 番目の要素」$\mathsf{enum}_R(2)$,... というふうに枚挙できるということである（ただし重複を許す）。それゆえ自然数 n が与えられて $R(n)$ が成り立つことを確かめるには、この枚挙を続けていきなんらかの m について $\mathsf{enum}_R(m) = n$ が成り立つことを示せばよい。しかしこのような枚挙をいくら続けても $R(n)$「でない」ということは確認することができない。例え 100000000 までこの枚挙を続けて n が現れなかったとしても 100000001 番目に n が現れる可能性は捨てきれないからである。このため、再帰的枚挙可能集合はしばしば**半決定可能** (partially decidable) 集合とも呼ばれる。

命題 4.22 *R, S が再帰的枚挙可能関係のとき、$R \wedge S$, $R \vee S$, $\forall y < z R(\vec{x}, y)$, $\exists y < z R(\vec{x}, y)$, $\exists y R(\vec{x}, y)$ も再帰的枚挙可能関係である。*

証明 $R \wedge S$, $R \vee S$, $\forall y < z R(\vec{x}, y)$, $\exists y < z R(\vec{x}, y)$ が再帰的枚挙可能関係となることは、命題 3.13、命題 3.15 と同様にして証明することができる。

$\exists y R(\vec{x}, y)$ については、部分再帰的関数 $\chi'_{\exists y R}$ を次のように定義すればよい：

$$\chi'_{\exists y R}(\vec{x}) = \mathsf{pos?}((\mu y R(\vec{x}, y)) + 1).$$

ここで $\mu y R(\vec{x}, y)$ が部分再帰的関数であることは明らかなので、$\chi'_{\exists y R}$ も部分再帰的関数である。 ■

上の性質は $\neg R$, $\forall y R(\vec{x}, y)$ については成り立たない。

定理 4.23 *$R \subseteq \mathbb{N}^n$ が再帰的関係であるための必要十分条件は、R と $\neg R$ がともに再帰的枚挙可能関係となることである。*

証明 R と $\neg R$ が再帰的枚挙関係であるとする。すると命題 4.21 により、

$$\begin{aligned} R(\vec{x}) &\iff \exists y(\mathsf{enum}_R(y) = \langle \vec{x} \rangle) \\ \neg R(\vec{x}) &\iff \exists y(\mathsf{enum}_{\neg R}(y) = \langle \vec{x} \rangle) \end{aligned}$$

が成り立つ。いま、$S(\vec{x}, y) \iff \mathsf{enum}_R(y) = \langle \vec{x} \rangle$、$T(\vec{x}, y) \iff \mathsf{enum}_{\neg R}(y) = \langle \vec{x} \rangle$ と書くことにする。排中律により $R(\vec{x}) \vee \neg R(\vec{x})$ は常に成り立ち、また

$$\begin{aligned} R(\vec{x}) \vee \neg R(\vec{x}) &\iff \exists y S(\vec{x}, y) \vee \exists y T(\vec{x}, y) \\ &\iff \exists y (S(\vec{x}, y) \vee T(\vec{x}, y)). \end{aligned}$$

であるから、$\exists y(S(\vec{x}, y) \vee T(\vec{x}, y))$ も常に成り立つ。ゆえに $\mu y(S(\vec{x}, y) \vee T(\vec{x}, y))$ は再帰的関数である。そして

$$R(\vec{x}) \iff S(\vec{x}, \mu y(S(\vec{x}, y) \vee T(\vec{x}, y))$$

がいえるので、R は再帰的関係である。 ■

命題 4.20 により、再帰的関係は全て再帰的枚挙可能関係であることが示された。では逆はどうだろうか？すなわち再帰的枚挙可能関係はすべて再帰的関係であるといえるだろうか？答えはノーである。このことを示しておこう。

前節で論じたように、1 項部分再帰的関数は、1 番目の関数 $f_0(x)$、2 番目の関数 $f_1(x)$、3 番目の関数 $f_2(x)$ ... と全部枚挙することができる。しかも x 番目の 1 項部分再帰的関数の指標を返す関数 $\mathsf{parrecindex}(x)$ は原始再帰的関数である。このことから、次のような性質を考えることができる。

> $\mathsf{defined?}(x,y) \Leftrightarrow x$ 番目の 1 項部分再帰的関数はインプット y において定義されている。すなわち $f_x(y)\downarrow$。

命題 4.24 $\mathsf{defined?}(x,y)$ は再帰的枚挙可能関係である。

証明

$$\begin{aligned}
\mathsf{defined?}(x,y) &\iff f_x(y)\downarrow \\
&\iff \mathsf{U}(\mu z \mathsf{T}_1(\mathsf{parrecindex}(x),y,z))\downarrow \\
&\iff \mu z \mathsf{T}_1(\mathsf{parrecindex}(x),y,z)\downarrow \\
&\iff \exists z.\mathsf{T}_1(\mathsf{parrecindex}(x),y,z)
\end{aligned}$$

T_1 は原始再帰的関係であるから、命題 4.22 により、$\exists z \mathsf{T}_1(\mathsf{parrecindex}(x),y,z)$ は再帰的枚挙可能関係である。 ■

命題 4.25 $\mathsf{defined?}(x,y)$ は再帰的関係ではない。

証明 $\mathsf{defined?}(x,y)$ が再帰的関係であると仮定して矛盾を導く。$\mathsf{defined?}(x,y)$ が再帰的関係ならばその否定 $\neg\mathsf{defined?}(x,y)$ も明らかに再帰的なので、次のように定義される関数 g は 1 項部分再帰的関数となる：

$$g(x) = \mu z(\neg\mathsf{defined?}(x,x) \wedge z = 1).$$

今、この関数が k 番目の 1 項部分再帰的関数 f_k であるとする。すると

$$\begin{aligned}
\mathsf{defined?}(k,k) &\iff f_k(k)\downarrow \\
&\iff \mu z(\neg\mathsf{defined?}(k,k) \wedge z = 1)\downarrow \\
&\iff \neg\mathsf{defined?}(k,k)
\end{aligned}$$

となり矛盾する。ゆえに $\mathsf{defined?}(x,y)$ は再帰的関係ではない。 ■

証明から明らかなように、実際には次のことが成り立つ。

系 4.26 $\mathsf{K}(x) \iff \mathsf{defined?}(x,x)$ により定義される性質 K は再帰的枚挙可能であるが、再帰的ではない。

以上をまとめると、次の関係が成り立つ：

(再帰的関係のクラス) = (再帰的枚挙可能関係のクラス)∩(再帰的枚挙可能関係のクラス)^C

(再帰的関係のクラス) $\subsetneq$ (再帰的枚挙可能関係のクラス)

ここで X^C は X の補集合をあらわす。

4.7 再帰的関数論の歴史的背景

以下は次の論文を参考にしている。

> Robin Gandy, The Confluence of Ideas in 1936, *The Universal Turing Machine: A Half-Century Survey (2nd. ed.)*, 1995.

決定問題を巡って：

- D. Hilbert(1900):「２０世紀に取り組むべき２３問題」の第１０番目：

  > ディオファントス方程式が与えられたとき、整数解を持つか否かを**有限回の操作により判定する方法**を見つけよ。

- M. Dehn (1911), A. Thue (1914): 群論における語の問題
- H. Behmann (1922): 高階論理を含む論理一般についての決定問題
- D. Hilbert (1928): 一階述語論理の決定問題（Entscheidungsproblem）:

  > （一階述語論理の）論理的表現が与えられたとき、それが恒真である（あるいは充足可能である）かどうかを**有限回の操作により判定する手続**は存在するか？

「有限回の操作により判定する手続」とは何か? 例えば最後の問題について言うならば、論理式の恒真性を判定する手続が存在するのならば、そのような手続を一つ具体的に提示すれば事は足りるが、もしもそのような方法がないのだとしたら、「ない」ということを数学的定理として示すためには、「有限回の操作により判定する方法」とは何かを数学的に定義する必要が出てくる。

不完全性定理を巡って：

- K. Gödel (1931,1934)：第一不完全性定理（の一般化されたヴァージョン）

  > 原始再帰的関数がすべて（数値ごとに）表現可能であるような**形式的体系** T をについて、もしも T が無矛盾ならば T には証明可能でも反証可能でもない論理式が存在する。

「形式的体系」とは何か？大雑把にいって、与えられた記号表現が公理であるかどうか、あるいは正しい推論規則の適用になっているかどうかが、「有限的に判定可能である」ような論理体系のこと。では「有限的に判定可能」とは一体どういうことか？$\Longrightarrow$「有限的に判定可能」の一般的定義へ。

原始再帰的関数：

- R. Dedekind (1888): 関数の帰納的定義
- T. A. Skolem (1923): 原始再帰的関数の研究
- D. Hilbert (1926): 高階の原始再帰法（of finite types）
- W. Ackermann (1928): 原始再帰的ではないが、二重原始再帰法（あるいは高階の原始再帰法）を使えば定義できる関数（アッカーマン関数）の発見

アッカーマンの結果により、少なくとも原始再帰的関数だけでは「実効的に計算可能」（Herbrand 1931）な関数全てを特徴付けるのには足りないことがわかる。

実効的に計算可能な関数の定式化：

- J. Herbrand (1931): ゲーデルへの手紙
- A. Church (1932): ラムダ計算の導入（プリンストン在籍）
- K. Gödel (1934): Herbrand-Gödel 再帰性の定義（不完全性定理の一般化に向けて。プリンストンでの講義にて）
- A. Church (1934):「実効的計算可能性＝ラムダ定義可能性」としようという提案
- S. C. Kleene (1936):「Herbrand-Gödel 再帰性＝ラムダ定義可能性」の証明。（現在普通使われている形の）一般再帰的関数の定義。後者も前二者と一致する。
- A. Church (1936): **チャーチのテーゼ**

  > 実効的に計算可能な関数とは、再帰的関数のことに他ならない。

チャーチのテーゼは本当に妥当か？だとしたらどのように正当化することができるのか？

機械的手続の分析：

- A. M. Turing (1936): テューリング機械の概念の導入。

  > 本当の問題は（「何が計算可能な関数か」ではなく）「数を計算するときに起こりうるプロセス（すなわち機械的手続）とは何か」である。

テューリングは、数学者が紙と鉛筆を用いて計算を行う過程を極限まで抽象化することによりテューリング機械の定義に到達した。(ちなみに現在使われている “computer” という言葉は彼の論文に由来するが、そこでは単に「計算する人」を意味するに過ぎない。) しかもテューリングは、テューリング機械により計算可能な関数はラムダ定義可能な関数と一致することも (ほぼ) 証明している。よってチャーチのテーゼは二重の意味で補強されることになった。第一に、全く違う立場からの定義がラムダ定義可能性＝再帰性と一致したということ。これは偶然とは考えにくい。第二に、テューリングの定義は人間が行う具体的な計算行為の徹底的な分析に基づいていること。これにより再帰性の概念が人為的なものではなく、現実に行われている計算に根ざすものであることが保証されたのである。

かくして「実効的に計算可能である」という直感的な概念は「再帰的である」(あるいはラムダ定義可能である、テューリング機械で計算可能である) という数学的な定義によって置き換えられうるということが広く認められるようになったのである。

- E. Post (1936)
- A. Church (1936), A. M. Turing (1936) : ヒルベルトの決定問題の否定的解決
- Y. Matiyasevich (1970): ヒルベルトの第 10 問題の否定的解決