# 再帰と帰納 :補足 1

八杉満利子

平成 14 年 5 月 27 日

## 1 テキスト 2 章の概要

2 章のうち、この科目でとりあげるのは 2 ・1 の [1]–[3]、2 ・4 の [1]、2 ・6 [1],[5] のみである。なお、2 ・3 の [1],[2] も参考になる。(これらの節の導入部分も読むこと。)

2 ・4 の [2]、8 6 ページの累積帰納法と 2 ・6 の [6] も読み物として読んでおくとよい。

この章で学んでほしいのは、次の 3 点である。

(1 ) 再帰的構造、再帰的定義という考え方があること、それらの例が命題論理やプログラムなどすでに知っている知識の範囲に多くあること。

(2 ) 自然数の関数で計算可能な関数とよばれる関数族を再帰的関数という形で正確に定義できること、ふつうに計算している関数はすべて再帰的であること。

(3 ) 数学的帰納法以外にいくつかの形の帰納法があり、それらは再帰的構造に関する命題の証明に有効であること。

どれも詳しいことは必要でない。基本的な概念をいくつかの例を知っていれば、必要なときに勉強できる。

なお、再帰的構造や構造帰納法の例は今後も出てくるので、注意してきいておいてほしい。

## 2 再帰的構造と再帰的定義の例

再帰的に定義されたオブジェクトの全体は再帰的構造である。

再帰的構造

1. (命題論理 ・述語論理の ) 論理式
2. (命題論理 ・述語論理の形式的 ) 証明
3. Unix のディレクトリの木構造
4. + と × と括弧で作られる式 (項 )

5. プログラム
6. 回文
7. 再帰的関数

再帰的定義

1. (命題論理の) 論理式の真理値
2. (命題論理の論理式の) 形式的証明

## 3 計算可能な関数とは

入力値も出力値も自然数である関数 (数論的関数という) が計算可能である、とはどういうことか ?おおまかにいえば、その関数の計算のためのプログラムが書けるような関数、といえる。正確な定義として、再帰的関数、Turing・マシン、ラムダ計算系、レジスタ ・マシンなどいろいろあるが、すべて同値である。すなわちそれぞれの意味で計算可能な関数の全体はどれでも同じである。ここでは再帰的関数をとりあげる。

再帰的関数は定数関数、恒等関数、「次の数」関数、を基礎にして、関数の合成、原始帰納的法、最小値という操作を繰り返して定義される。 (p.75) これらはどの計算機言語でも定義できるので、実行してみるとよい。

「ふつうの」数論的関数は再帰的関数である。たとえば、$x+y$, $x \times y$, $x^y$, $x!$ など。 $x+y$ の計算方法の概要を示す。$x+y$ を $plus(x,y)$ と表す。$id(b)=b, S(c)=c+1$ とする。まず $add(a,b)$ を定義する。

$add(a,b) = \text{if } a=0 \text{ then } id(b) \text{ else } S(add(a-1,b))$

$$plus(x,y) = add(y,x)$$

これが実際 $x+y$ であることを見よう。

$$plus(x,0) = add(0,x) = id(x) = x = x+0$$

$$plus(x,y+1) = add(y+1,x) = S(add(y,x)) = add(y,x)+1$$

$$= plus(x,y)+1 = (x+y)+1 = x+(y+1)$$

演習問題 1 . $plus$ を使って、$mult(a,b) = a \times b$ となるように $mult$ を再帰的に定義し、実際に $mult(a,b) = a \times b$ になることを示せ。

2 . $mult$ を使って、$exp(c,d) = c^d$ となるように $exp$ を再帰的に定義し、実際にこの等式がなりたつことを示せ。

3 . $mult$ を使って $fuct(c) = c!$ となるように $fuct$ を再帰的に定義し、実際にこの等式がなりたつことを示せ。

以下の問題は再帰的定義にこだわらず、各自好みのプログラム言語で定義してみるとよい。 (自然数の入力の場合にのみ等式を満たせばよい。)

4 ．$minus(c,d) = c - d(c \geq d); = 0(c < d)$ となるように $minus$ を再帰的に定義せよ。

5 ．$x \geq y \Leftrightarrow \chi(x,y) = 1; x < y \Leftrightarrow \chi(x,y) = 0$ となるような $\chi$ を再帰的に定義せよ。