# 計算機がはいった頃の思い出

平　口　俊　夫

本学にＮＥＡＣが導入されたのは１９６３年であったと記憶する。同年８月machine codeの手ほどきを受けると，早速計算機ととり組んだ。当初はまだ計算機もひまであったから，操作卓にへばりついて自製のプログラムの進行状況を自ら看守することができてたのしみであった。いわばずぶの素人がいきなり，やみくもに計算機に取り組んだのだから，プログラミングに際して犯しやすい誤りは始ど経験したことはいうまでもない。このような初歩的な知識のみならず，計算機から啓蒙されて，かくされていた数学の問題にはじめて気がつくといった事例もあり，計算機から教えられることが多かった。

当時私は〝6個の元の集合の上のpartal ordersのうちdimensionが３のもの(それをrearizeするのに３個のlinear ordersを必要とするもの)は２つ(isomorphおよびdual isomorphを除いて)しかない〟こと，また〝7個の元の集合の上のPartial ordersのうちdimension 3のものは，上の２つのpartial ordersのいづれかをその部分として含む〟という２つのconjecturesをもっていた。しかしその真偽数学的証明となると，まるで見当がつかず，手掛りすら摑めないまゝに放置されてあった。考えられるたゞひとつの手段は，可能なあらゆる,partial ordersをgeuerateして、ひとつひとつそのdimensionをしらべてみるという，いわばeumeration法である。しかし数多くのpartial orderを洩れなくgenerateして，そのddimensionを評価していくことは，人力でははなはだ心もとなく，これは正に電子計算機にうってつけの仕事であった。しかもはじめに２つのlinear ordersによってraalizeされるPartial ordersの異なるものをlist upしておいて，３つのlinear ordersのうち上のlistにのっていないものだけをout putさせればよいのだから原理的にははなはだ単純な仕事である。もっとも実際には，主記憶装置の容量からくる制約と，所用時間をできるだけ短縮したいという要請のため，色々面倒なことがある。

２個の要素の上のlinear ordersは，２個の要素を１から２までの自然的であらわすことにすれば，１から２までの数の順列にほかならないから，何よりもまず２個の数の順列を発生させるプログラムを組む必要がある。最初は１としての順列から，その各に３を挿入することによって，１，２，３の順列をつくり，さらにその各々に４を挿入して，１，２，３，４の順列をつくるといったようにして，１から６までの数の順列をつくり，これを昇順にならべて記憶装置に格納しておき，その上を繰り返し走査する方法をとった。順列の作り方の拙劣さはさておき，このやり方ではｎ＝７の場合は，ＮＥＡＣ－２２３０の内部メモリの容量をはるかに超えてしまうので適切でない。できれば順列を辞書式順序に，必要に応じて次々と発生させることが望ましい。そのためには与えられた順列から次の順列を発生させるプログラムをつくることができればよいわけである。ああでもない，こうでもないと摸索を繰返しているうちに，ふと隣接する順列の間の関係をしらべてみることにした。このことに気がつくことの何とおそかりしことよ！

１からｎまでの数の順列のひとつを $a_1 a_2 \cdots a_n$，辞書式順列でその次の順列を $b_1 b_2 \cdots b_n$とすれば，次の条件を満足する数ｊが存在することが証明されるのである。

1）　$a_1 = b_1$，$a_2 = b_2$，…，$a_{j-1} = b_{j-1}$，$a_j < b_j$

2）　$a_j < a_{j+1}$，$a_{j+1} > a_{j+2} > \cdots > a_n$

3）　$b_j > b_{j+1}$，$b_{j+1} < b_{j+2} < \cdots < b_n$

4）　$b_j = \min\{a_k \mid a_k > a_j\}$

このことから次の手順によって順列 $a_1 a_2 \cdots a_n$ の次の順列 $b_1 b_2 \cdots b_n$ をつくることができる。

1　条件２）を満たす数ｊを見出す。

2　$a_k > a_j$ なる $a_k$ のうちは最小のものを見出し，これを $b_j$ とする。

3 $a_j$ , $a_{j-1}$ , …, $a_n$ の中から$b_j$を除いたものを昇順にならべかえたものを $b_{j+1}$ , $b_{j+2}$ , …, $b_n$とする。

4 $a_1 a_2 \cdots a_{j-1}$ を改めて $b_1 b_2 \cdots b_{j-1}$ とすれば

$b_1$ , $b_2$ , …, $b_{j-1}$ , $b_j$ , $b_{j+1}$ …, $b_n$ が求める順列である。

これによってプログラムを組むことは容易であろう。順列を辞書式順序で発生させる問題はこれで完全に片がついた。この方法は順列を発生させるもっとも能率的な方法であろう。辞書式順列で隣り合う２つの順列の比較検討を最初にやっておけば，無駄な暗中模索によって時間を費やすことはなかったのである。いまから考える誠にだらしない話である。

なおこれは後になって知ったのであるが，Mathematial Reviews Vol May No，５，May 1９６５の頁をめくっていたら，偶然S．Mok－koug：On the generation of permutation and combination(１９６２）の紹介記事が目についた。原論文は見ていないが，その記事によると，Mok－Kong氏の方法は上記の方法と全く同じものらしい。同じ頃同じような問題を考える人がいるものである。

# 計算機械学への第一歩

木　戸　睦　彦

うかつなことながら，計算機科学(computerScience)という言葉があることを，つい最近まで知らなかった。知ってみると，そう多くはないが，あちこち目につく。もっとも，その定義の明瞭でないことは，情報科学などという言葉と同様である。ハードのことも，ソフトのことも，とにかく計算機に関係あるすべてのことはみな計算機科学に含めてもよいらしい。一方で計算機科学というものに対する疑問もあるらしい。そもそも科学とは，現象について記述したり説明したりするものであるが，計算機は人工的機械で一定の法則に従うというわけのものではないから，計算機科学などというものはあり得ないのではないか，などというのである。

計算機科学が何であるかはともかくとして，我々は中型の優秀な計算機をもつことになった。もし，これが会社に導入されたもので，我々がその会社から，計算機の使用法についてアドバイスを求められたらどんなことを言うだろうか。今迄人手でやっていた給与計算や料金計算を計算機にやらせただけで事足れりとするなら，計算機導入の意義はない。各部課が思い思いに使うだけでなく，会社的視野でＭＩＳとかＯＲとか新しい手法を用いて意志決定をするために計算機を活用するのでなければメリットは生れない。というようなことを言うだろう。

大学は会社のようなメリットを考える必要はない。けれども研究と教育のために計算機を有効に使うべきであるということでは，我々もまた我々のメリットに意を用いねばなるまい。手でやるのは大変だ，という計算を計算機にやらせるばかりなら，計算機のソロバン化で，会社が給与計算や料金計算だけやっているようなものである。我々が会社に対してなすであろう忠告は我々自身に向けてもなさねばなるまい。

ここに，定義は不明ながら計算機科学というようなものを求める必要が起ってくる。それが科学の名に値するかどうかなどということはどうでもよく，計算機の能力をどのようにして生かすかを探し，その知識の交換と未来のユーザーへの伝達を考えねばならない。会社で会社的視野に立つ必要があるように，大学では学部学科の壁を除いて，セミナーや講義が行われねばならない。今のところ，その音頭をとるのはセンターであろう。カリキュラムの確立などは大事業であろうが，ともかく踏み出す第１歩としてなら，今迄のユーザーは講義してよいような内容の１つや２つは持ち合せていると思うのである。