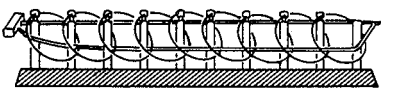
CHINA RING 考

植　松　茂　暢

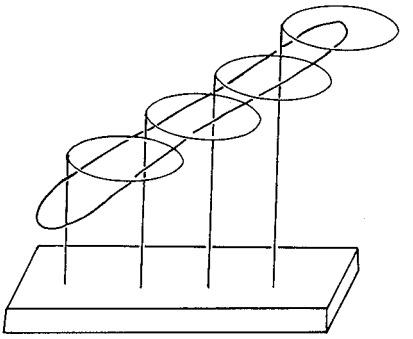
第1図の九連環は，中国で2000年以上昔に作られ，西洋に伝わり CHINA RING と呼ばれ，日本へは17世紀ごろには既に伝わっていたという。現在も同類の玩具が多く作られ市販されている。第1図の九連環は第2図の環を9個連ねたものと同じと考えられるので，CHINA RING としては第2図の型のものをとり，問題は第3図の状態から紐を RING に通すことにより第4図の状態に変えることであるとします。



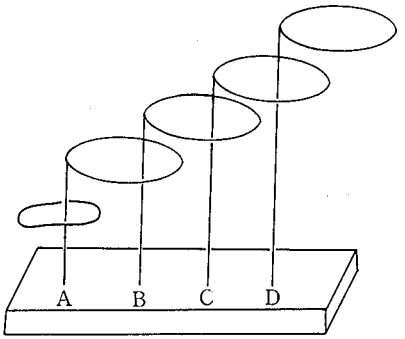
第1図

CHINA RING の解法については，古くから研究され，この一般教育研究第18号においても小林氏によって論じられています。

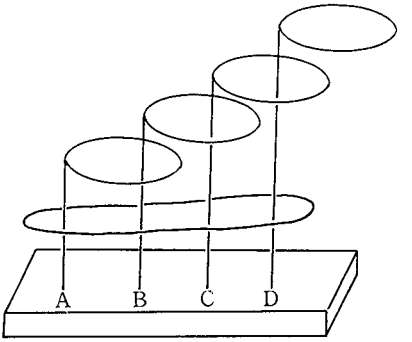
ここでは，CHINA RING の解法をマイコンのプログラミングの立場から考察する。



第2図



第3図



第4図

1. 解法の Tactics について

CHINA RING の解法を環の数が n=2, n=3 の場合など簡単な場合について考察すると，解法はつぎの4個の Tactic から成ることが帰納的に容易にわかる。

（1） UP して COVER (UC)

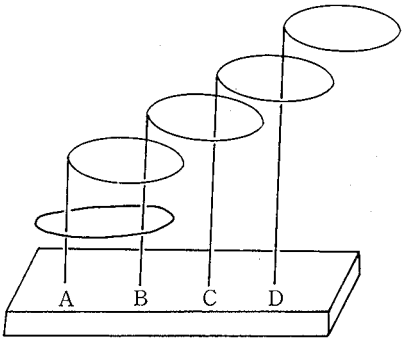
第5図の状態から第6図の状態に変える操作

（2） UNCOVER して DOWN (UD)

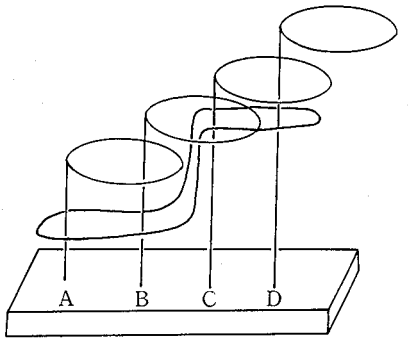
操作 UC の逆操作で，第6図の状態を第5図の状態に変える操作

（3） UP して SEPARATE (US)

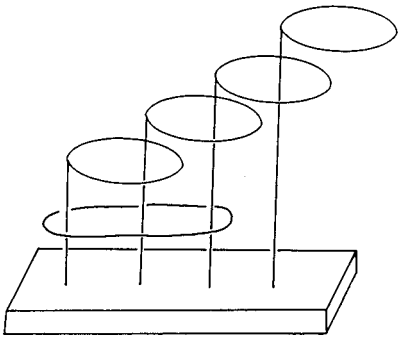
第7図の状態から第8図の状態に変える操作



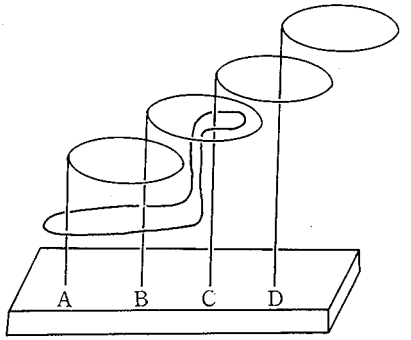
第5図



第6図



第7図



第8図

（4） DOWN して LEVEL (DL)

操作 US の逆操作で，第8図の状態を第7図の状態に変える操作

2. 解法の Strategy

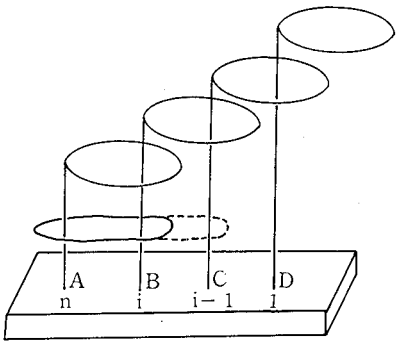
CHINA RING の問題解決には，第9図のように，i 番目の棒まで囲んだ紐を i−1 番目の棒まで囲むようにすることが基本で，これを n から n−1, n−2, ⋯, 2 と進めるとよい。そこでつぎの2つの操作を定義する。

（1） STEP FORWARD (SF_i)

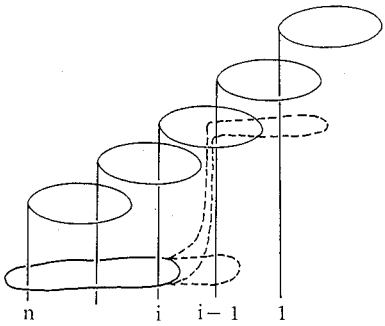
第9図のように，棒の囲みを一歩前進させる操作

（2） STEP BACK (SB_i)

操作 SF の逆操作で，棒の囲みを一歩後退させる操作



第9図



第10図

第9図で操作 SF_i を行うには，RING i について操作 UC を行ない，つぎに囲いを1番目の棒から2番目の棒に後退させる SB_2，2番目から3番目に後退する SB_3，⋯，SB_{i-1} を行ない，最後に DL をすればよい。すなわち

$$SF_i = UC + SB_2 + SB_3 + \cdots + SB_{i-1} + DL_1 \quad \cdots\cdots ①$$

以上のことを逆に見れば，i−1 番目の棒まで囲んだ紐を i 番目の棒まで囲むように後退させる操作 SB_i になるから

$$SB_i = US + SF_{i-1} + SF_{i-2} + \cdots + SF_2 + UD \quad \cdots\cdots ②$$

となる。CHINA RING の解法は第3図を第4図に変えることであるから，

RING の数が n 個のときの解法は

$$SOL_n = SF_n + SF_{n-1} + \cdots + SF_2 \quad \cdots\cdots ③$$

である。

Strategy ①, ②, ③のうち①, ②は再帰的定義であって FORTRAN 言語, ALGOL 言語などではプログラミングすることはできない。しかし LISP 言語では, この定義をこのままの形でプログラミングすることができる。第11図は環の数が4個のとき, LISP 言語のプログラムを実行した結果である。

```
(CHINA 4)

0    (INITIAL)                  (A) B C D
1    (UP AND COVER)             ((A) B C D)
2    (UP AND SEPARATE)          (((A) B C) D)
3    (UNCOVER AND DOWN)         ((A) B C) D
4    (UP AND SEPARATE)          (((A) B) C) D
5    (UP AND COVER)             ((((A) B) C) D)
6    (DOWN AND LEVEL)           (((A) B) C D)
7    (UNCOVER AND DOWN)         ((A) B) C D
8    (DOWN AND LEVEL)           (A B) C D
9    (UP AND COVER)             ((A B) C D)
10    (UP AND SEPARATE)          (((A B) C) D)
11    (UNCOVER AND DOWN)         ((A B) C) D
12    (DOWN AND LEVEL)           (A B C) D
13    (UP AND COVER)             ((A B C) D)
14    (DOWN AND LEVEL)           (A B C D)
NIL
*
```

第11図

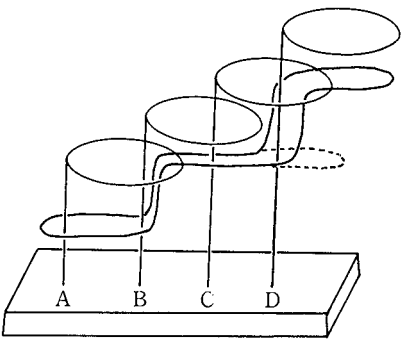
第11図について説明する。

(A)BCD

は, A の環の棒にのみ紐がかかっている INITIAL の状態を示している。また

(((A)BC)D)

は, 第12図のように A の環の棒を囲んだ紐が A の環を通って上り D の環の棒まで囲んだ（点線の部分）状態から, 操作 US によって C の環を通して紐を

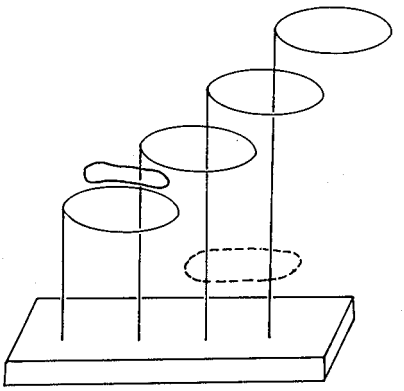


第12図

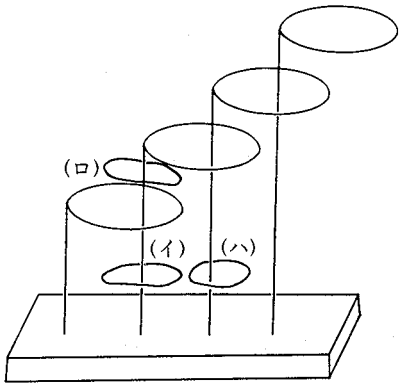
上にあげた状態になったことを示している。他は類推できるであろう。

3. もう1つの Strategy

CHINA RING の解法のもう1つの Strategy は，第13図の実線の状態を REVERSE (REV_i) して点線の状態に変える操作を用いる方法である。



第13図



第14図

第14図の実線のようにi番目の棒を囲んだ紐（ロ）の状態を i−1 番目の棒を囲んだ紐（ハ）の状態に変化させる操作を TRANSFER (TR_i) とすると

$$TR_i = REV_i + SB_2 + SB_3 + \cdots + SB_{i-1} \quad \cdots\cdots ④$$

と表わすことができる。i 番目の環の棒を囲む紐（イ）を i−1 番目の環の棒を囲む紐（ハ）の状態に変化させる操作は，i≒n のときは，i+1 番目の環を通して LIFT しなければならないので，CHINA RING の解法は

$$SOL_n = TR_n + (LIFT + TR_{n-1}) + \cdots + (LIFT + TR_2) + LIFT \quad ⑤$$

となる。この strategy で LISP 言語でプログラミングして実行した結果が第15図である。REVERSE を回数に入れていないのは，紐が環の中をくぐらないからである。

```
(CHINA 4)

0    (INITIAL)                (A) B C D
     (REVERSE)                A (B C D)
1    (UP AND SEPARATE)        A ((B C) D)
2    (UNCOVER AND DOWN)       A (B C) D
3    (UP AND SEPARATE)        A ((B) C) D
4    (UP AND COVER)           A (((B) C) D)
5    (DOWN AND LEVEL)         A ((B) C D)
6    (UNCOVER AND DOWN)       A (B) C D
7    (LIFT)                   A (B) C D
     (REVERSE)                A B (C D)
8    (UP AND SEPARATE)        A B ((C) D)
9    (UNCOVER AND DOWN)       A B (C) D
10    (LIFT)                   A B (C) D
      (REVERSE)                A B C (D)
11    (LIFT)                   A B C (D)
NIL
*
```

第15図

4. 解法の手数について

多くの書物で説明している CHINA RING の解法は REVERSE を用いない方法であり，手数の計算もそれによっている。ここでも先づ，それによって説明しよう。

操作 SF_i の手数を $m(SF_i)$ で示し，他の操作の手数も，これにならうことにする。公式①，②より

$$m(SF_i)=1+m(SB_2)+m(SB_3)+\cdots+m(SB_{i-1})+1$$

$$m(SB_i)=1+m(SF_{i-1})+m(SE_{i-2})+\cdots+m(SF_2)+1$$

となり，また明らかに

$$m(SF_i)=m(SB_i)$$

であるから

$$\begin{aligned} m(SOL_n)&=m(SF_n)+m(SF_{n-1})+\cdots+m(SF_2)\\ &=2\{m(SF_{n-1})+m(SF_{n-2})+\cdots+m(SF_2)\}+2\\ &=2^2\{m(SF_{n-2})+m(SF_{n-3})+\cdots+m(SF_2)\}+2^2+2 \end{aligned}$$

$$= \cdots\cdots$$

$$= 2^{n-2}m(SF_2) + 2^{n-2} + 2^{n-3} + \cdots + 2$$

$$= 2^n - 2$$

このように解法の手数を計算したが，これは環の中を紐が通る回数を数えた手数である。操作 UC では UP するとき紐が環の中を通るので1回と数え，COVER するときは紐が環の外を通るので回数に入れていない。

同じ解法でも，操作 UC を UP で1回，COVER で1回，計2回の手数と数えると

$$m(SF_i) = 2 + m(SB_2) + m(SB_3) + \cdots + m(SB_{i-1}) + 1$$

$$m(SB_i) = 1 + m(SF_{i-1}) + m(SF_{i-2}) + \cdots + m(SF_2) + 2$$

となり，前と同様に計算して

$$m(SOL_n) = 3(2^{n-1} - 1)$$

となる。

REVERSE を用いたときの解法の手数は，紐が RING の中を通る回数のみ数えるならつぎのようになる。④より

$$m(TR_i) = m(SB_2) + m(SB_3) + \cdots + m(SB_{i-1})$$

$$= 2 + 2\{m(SB_2) + m(SB_3) + \cdots + m(SB_{i-2})\}$$

$$= 2 + 2^2 + 2^2\{m(SB_2) + m(SB_3) + \cdots + m(SB_{i-3})\}$$

$$= \cdots\cdots$$

$$= 2 + 2^2 + \cdots + 2^{i-3} + 2^{i-3}m(SB_2)$$

$$= 2 + 2^2 + \cdots + 2^{i-3} + 2^{i-2} = 2^{i-1} - 2$$

従って，解法の手数は⑤より

$$m(SOL_n) = (2^{n-1} - 2) + (1 + 2^{n-2} - 2) + (1 + 2^{n-i} - 2) + \cdots$$

$$+ (1 + 2^i - 2) + 1$$

$$= 2^n - n - 1$$

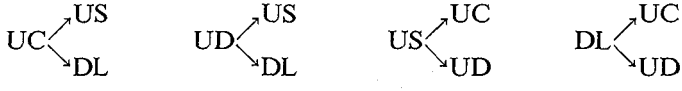
となる。このように，操作 REVERSE を用いる解法は多くの書物に記されている手数 2_n-2 より少い回数となる。

多くの書物には「CHINA RING の解法の手数は 2_n-2 である」ように誌されているが，その書き方はあたかも「それが最小手数である」かのように書かれているが，「CHINA RING は 2_n-2 回の手数で解ける」とすべきであろう。2_n-2 回が最小手数であることを証明することは容易でなく恐らく，すべての方法を片っ端しから調べるという方法でしか分らないこと，従って，一般には証明不可能な問題ではなかろうかと思う。

5. BASIC 言語による解法プログラミング

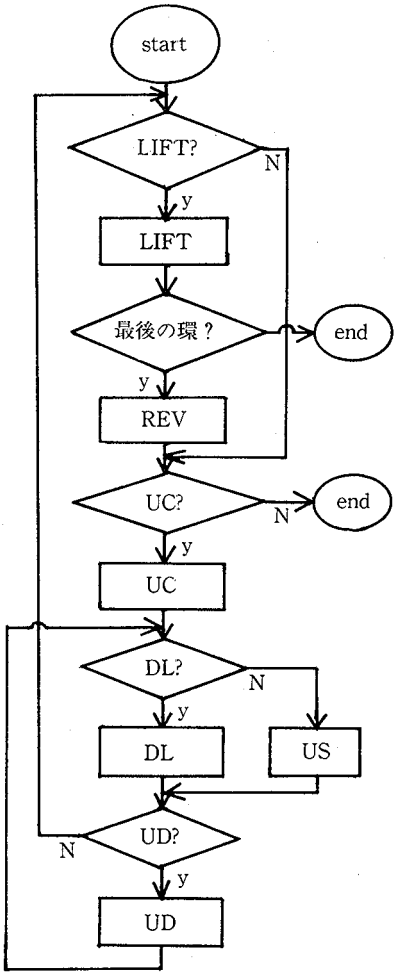
CHINA RING の解法を学生に BASIC 言語でプログラミングさせようと思って，BASIC 言語でもプログラミングしてみました。Strategy ①，②は BASIC 言語を用いてプログラムできないので，つぎのような Tactic を用いました。

操作 UC と UD，操作 US と DL は互に逆操作であるから，これらを続けて操作することは無意味である。したがって，続けて行なう操作はつぎのものだけである。



操作 US より DL を，操作 UC より UD を優先する Tactic をとります。操作 REV は，これを許すならば，可能な場合には，いつも優先します。このように考えて作ったフロチャートが第16図です。プログラムは簡単である。

最後に LISP 言語のシステムを提供し，説明をして下さった本田道夫氏（経済学部）に謝意を表記します。



第16図