組み合わせと魔方陣

SHIMURA Masato
http://japla.sakura.ne.jp

2014 年 10 月 27 日

目次

はじめに

マイケル ·J· ブラッドリー著の「数学を切りひらいた人びと」（全 5 巻　青土社) は時代区分毎に 10 人づつ 50 人の数学者を取り上げる。デカルト、コーシー、ベルヌイ一家、ラグランジュなどが抜けて、ナイチンゲール、シェルピンスキーやアラン · チューリング、COBOL のホッパー女史、コンウエイなど学校数学ではあまりお目にかからない近代や現代の数学者が取り上げられた偏った人選だが、現代数学の入門として興味深く読める。この 50 人の最後はアイルランドの早熟の少女セアラ · フラナリー（1982-) で、かのハミルトンを差し置いて選ばれている。セアラは 16 歳でマトリクスを秘密鍵に用いた高速の CP 暗号（ケーリー · パーサー暗号) を考案した。セレア自らの著書「16 歳のセアラが挑

んだ世界最強の暗号」NHK 出版がある。　　父はコーク工科大学の数学者でフラナリーに幼いころから興味を引く問題を台所の黒板に張り出していた。

1　魔方陣

父がセアラに出した問題の一つに 1 から 9 までの数字で作る 3×3 の魔方陣がある。魔方陣は縦、横、斜めの数字が同じになる。[*1]

3×3 のマトリクス

1	2	3
4	5	6
7	8	9

2　組み合わせ

- 組み合わせを作る J 言語のスクリプト。これを BASIC やったら多分目が回る。

```
tap=: i.@! A. i.
NB. table of permutations /組み合わせのテーブル
```

i.　数列を生成

```
   i.3
0 1 2
```

A.　*Anagram*　辞書式順序（組み合わせを順にすべて表示する）

```
    ! 3
6                        NB.  3*2*1
   i. ! 3
0 1 2 3 4 5              NB. i.6

   (i. ! 3) A. i.3       NB. (i.6) A. i.3  (=tap)
0 1 2
0 2 1
```

[*1] 魔法陣とは異なる

```
1 0 2
1 2 0
2 0 1
2 1 0
```

- 2 の場合　(0,1) の組み合わせは２つ

```
  tap 2
0 1
1 0
```

- 3 の場合 (0,1,2) の組み合わせは６とおり

```
  tap 3
0 1 2
0 2 1
1 0 2
1 2 0
2 0 1
2 1 0
```

- ４の場合 (0,1,2,3) の組み合わせは 24 とおり

```
  tap 4
0 1 2 3
0 1 3 2
0 2 1 3
0 2 3 1
0 3 1 2
0 3 2 1
1 0 2 3
1 0 3 2
1 2 0 3
1 2 3 0
1 3 0 2
1 3 2 0
2 0 1 3
2 0 3 1
2 1 0 3
2 1 3 0
2 3 0 1
2 3 1 0
3 0 1 2
3 0 2 1
3 1 0 2
3 1 2 0
3 2 0 1
3 2 1 0
```

- 書き上げるのは大変だから個数だけ数え上げよう。個数の数え上げは単なる階乗の計算である。
  1. ＃　個数 n を求める
  2. ！　階乗の計算
  3. NB. コメント

```
   # tap 4
24
   # tap 5
120
   # tap 6
720
   # tap 7
5040
   # tap 8
40320
```

- 9 の場合は 36 万をこえる。

```
   # tap 9
362880

  ! 9     NB. 9*8*7*6*5*4*3*2*1
362880
```

- このスクリプトは全ての組み合わせを作成したのち、個数を数えている。もう少しで内部メモリー２ GB のコンピューターがパンクする。組み合わせで結果をすべてメモリーに持たせる方法はコンピュータをいじめるのによい。

```
   # tap 11
|out of memory: tap
|   #    tap 11
```

2.1　組み合わせ論を少し

離散数学の組み合わせ論はインドの数学者バースカラ (1114-1185 頃) に始まり、南フランスのゲルソニデス（1288-1344) など様々な数学者が取り組み、ヤーコブ・ベルヌイ (1654-1705) の名著「推測の技術」でまとめられた。

1. !の単項 階乗/Factorial

$$!5 \longleftrightarrow 5 \cdot 4 \cdot 3 \cdot 2 \cdot 1 = 120$$

2. !の両項 組合せ/Combination

$$3!5 \longleftrightarrow_5 C_3 \quad = \frac{5 \cdot 4 \cdot 3 \cdot 2 \cdot 1}{(3 \cdot 2 \cdot 1)(2 \cdot 1)} = 10$$

$$_nC_r = \binom{n}{r} = \frac{n!}{r!(n-r)!}$$

3. 順列を用いると

$$_nP_r =_n C_r \cdot !r \longleftrightarrow_n C_r = \frac{_nP_r}{!r}$$

$$(3!5) * !3 \longleftrightarrow 5 \cdot 4 \cdot 3 = \frac{5 \cdot 4 \cdot 3}{3 \cdot 2} * 3 \cdot 2 = 60$$

3　3 × 3 の魔方陣

1 から 9 までの数字で魔方陣を作る。1 から 9 までの数字を 3 × 3 のマトリクスに組む方法は 362,880 あった。魔方陣は縦、横、斜めが全て同じ数になる。先ずはプログラムの力技で行こう。

```
   fm0 ''
┌-----------------------------------------------------------
| 2 7 6 | 2 9 4 | 4 3 8 | 4 9 2 | 6 1 8 | 6 7 2 | 8 1 6 | 8 3 4 |
| 9 5 1 | 7 5 3 | 9 5 1 | 3 5 7 | 7 5 3 | 1 5 9 | 3 5 7 | 1 5 9 |
| 4 3 8 | 6 1 8 | 2 7 6 | 8 1 6 | 2 9 4 | 8 3 4 | 4 9 2 | 6 7 2 |
-------------------------------------------------------------

   (0)     (1)     (2)     (3)     (4)     (5)     (6)     (7)
```

8 個の魔方陣が打ち出された。これを整理してみると次のようになる。(2) と (5) は 2 個の軸での作用が重なる。これは 右/左の $\frac{1}{4}$ 回転と同じである。

Axes		|	−	+
	8 1 6 3 5 7 4 9 2	6 1 8 7 5 3 2 9 4	4 9 2 3 5 7 8 1 6	2 9 4 7 5 3 6 1 8
↙	2 7 6 9 5 1 4 3 8	6 7 2 1 5 9 8 3 4	4 3 8 9 5 1 2 7 6	
↘	8 3 4 1 5 9 6 7 2	4 3 8 9 5 1 2 7 6	6 7 2 1 5 9 8 3 4	
×	2 9 4 7 5 3 6 1 8			

このように整理すると、次のようになる。

- 32 万の組み合わせの内、魔方陣は (6) の基本 1 種類とその派生が 7 パターンである。
- 行、列、対角の和は各 15 である。
- 5 は必ず、中心にくる
- 偶数は必ず、4 隅に来る

		|	−	+
	6	4	3	1
↙	0	5	2	
↘	7	2	5	
×	1			

ブラッドリーによると少女フラナリーは上の 3 の要素を掴んでいたようだ。32 万の組み合わせの中でパワースピリットのあるのはこの 8 個のみであると、何回も基本マトリクス (6) を動かして推論したのだろう。パズルの論理と数理に取り組むことでフラナリーの問題解決への技能が大いに育まれたとブラッドリーは書いている。

3.1 3×3 の魔方陣のスクリプト

```
    fm=: find_mahoujin=:3 : 0
NB. find Mahoujin  3 X 3
NB. Usage: fm ''
nr=. # tmp0=. >: tap 9            NB. 全組合せを求め 1 を加え 1-9 の整数に
ans=. < 0
for_ctr. i. nr do.                NB. カウンター付きで 32 万回のループ
 tmp2=. (+/;+/"1)   3 3 $ ctr{tmp0   NB. 縦横各行の合計
 tmp4=.  +/ L:0  (0 4 8;2 4 6){L:0 ctr{tmp0   NB. 斜め方向を取り出し合計
  if. 1 = # ~. ; tmp2,tmp4 do.    NB. ~. nub ８個が全部同じ数なら魔方陣
   ans=. ans,<ctr{tmp0            NB. 魔方陣の書き出し
  end.
end.
3 3 $ L:0 }. ans                   NB. 3x3 に整形
)
```

3.2 A. アナグラム

更なるアナグラム (辞書式順序）の機能を見ておこう

1. !9 → 362880
2. ９桁の順列の 1，2 番を A. で求める。

```
       1 2 A. i.9
0 1 2 3 4 5 6 8 7
0 1 2 3 4 5 7 6 8
```

3. 魔方陣の 8 組

```
>, L:0 a
2 7 6 9 5 1 4 3 8
2 9 4 7 5 3 6 1 8
4 3 8 9 5 1 2 7 6
4 9 2 3 5 7 8 1 6
6 1 8 7 5 3 2 9 4
6 7 2 1 5 9 8 3 4
8 1 6 3 5 7 4 9 2
8 3 4 1 5 9 6 7 2
```

4. 8組の組み合わせの辞書での位置を求めることができる

```
A.  >, L:0 a
69074 77576 135289 157120 205759 227590 285303 293805
```

3.3 魔方陣の固有値

魔方陣の固有値をルベリエ · ファディーエフ法で求める。同類の 8 個の対称型は複素数の有無で 2 種類に分かれる

```
,.> a1=. 1{"1  >  char_lf L:0 a
15   4.89898   _4.89898
15   4.89898   _4.89898
15 0j4.89898 0j_4.89898
15 0j4.89898 0j_4.89898
15 0j4.89898 0j_4.89898
15 0j4.89898 0j_4.89898
15   4.89898   _4.89898
15   4.89898   _4.89898
```

4　高次の魔方陣

魔方陣は 1 から n^2 の数を用いて、行、列、対角の和は次になることが知られている。[*2]

$$\frac{1}{n}\sum_{i=1}^{n^2} i = \frac{n(n^2+1)}{2}$$

tap 16,tap25,tap36 などの A. を用いる tap はメモリが厳しいので、逐次打出しのアナグラムを自作しなければならない。

デューラー (ドイツ）の 1514 年の銅版画に 4×4 の魔方陣が描かれている。

WikiPedia によると 4×4 の魔方陣は 880、5×5 は 2 億 3750 万個存在することが知られている。（以降の個数は記されていない。）

WikiPedia にはさらりと書いてあるが、力技法では 4×4 の魔方陣は!16=21 兆回のループを要する。

```
      !16
2.09228e13
   !25
1.55112e25      NB. 15 Jyo
```

4×4 の最初の方の魔方陣でもループ 809 億回。全て求められた方々に敬意を表する。

```
      A. 1 2 15 16 13 14 3 4 7 12 10 5 8 11 6 9
80907739690
```

1	2	15	16
13	14	3	4
12	7	10	5
8	11	6	9

関孝和、オイラー、ラマヌジャン、コンウエイも数論として魔方陣を研究している。

[*2] 他のタイプの魔方陣もある

付録 A　高次の魔方陣のスクリプト

- メモリーが厳しいので、ループで一個ずつ 魔方陣か否か判定して、魔方陣ならば書き出す。
- `for_ctr. i. 1000000000 do.` とするとメモリーオーバーになる。i. の全ての数をメモリーに持つため。while. ループに変更した。(メモリはあまり使わない)

```
 fm0=: find_mahoujin_any =: 4 : 0
NB. find Mahoujin  3X3 4X4 5X5 ...
NB. Usage: (0, 10000) fm0 3 3
NB. Usage:( 80907762000,80907762100) fm0  4 4
NB. x is pickup zone of permutiation
NB. i.e. 3 8 is 3<->8
NB. ! 9 16 25 is 362880 2.09228e13 1.55112e25
NB. y is i.e 3 3  // 3 X 3
NB. -------------------------------
'nrmin nrmax'=. x                        NB. input zone //oligin is 0
zone=. (<: nrmax) - nrmin
if. 2 = # y do. size0=. y
 else. size0 =. 2 # y
end.
size=. */ size0                          NB. i.e. 9 <-- 3 X 3
NB. -------------------------------
mat=. i. size0                           NB. explore index  matrix
oblic_ind=. index_oblique mat            NB. find oblic index
ans=. < 0
ctr=. nrmin
NB. -------------------------------
while. ctr < nrmax  do.
 mmat=.  >: ctr A. i. size              NB. target permutation  1++
 tateyoko=. (+/;+/"1)  size0 $ mmat       NB. make mat and sum tate&yoko
 naname=. +/ L:0 oblic_ind { L:0  mmat    NB. sum each oblic X
```

```
  if. 1 = +/ ~: ; tateyoko,naname do.     NB. NB. 4 pieces is same --> Mahoujin
      ans=. ans,<mmat
  end.
ctr=. >: ctr
end.
NB. ------------------------------
if. 1 = 1 < #  ans do. ans=. size0 $ L:0 }.ans
else. ans=. 'nothing'                       NB. null
end.
)

 index_oblique=: 3 : 0
NB. sub find oblic index
NB. usage: index_oblic mat
NB. y is mat
ob0=. ((<:# y) {  </.    y)
ob1=. ((<:# y) {  </. |. y)
oblique=. ob0,ob1      NB. pick index both / & \
)
```

付録 B　ルベリエ・ファディーエフ法のスクリプト

```
NB. ========Eigenvalue========================================
NB. find Eigenvalue and Eigenvector
NB. Levierre Faddeev Method
NB. -------------------------------
tr=: (<0 1)&|:
NB. umatrix=: (=/~)@i.@#
char_lf=: 3 : 0
```

```
ANS=.TR_SUM=. +/ tr MAT=. y NB. sum of trace
UMAT=. =/~ i. # y
for_LF. i.<: # y do.
MAT=. y +/ . * MAT - UMAT *  TR_SUM
TR_SUM=. (% 2+LF)* +/  tr MAT
ANS=. ANS,TR_SUM
end.
(p. POL), (<POL=.(-|. ANS),1)
)

char_lf_evec=: 3 : 0
EIGEN=. {@> ; 1{ char_lf y
EIGEN2=. {@> L:0 EIGEN ^/ L:0 |. i. # EIGEN
ADJMAT=. char_lf_evec_sub y
ANS=. <''
for_LF. i. # y do.
TMP=. +/> ( > LF{ EIGEN2) * L:0 ADJMAT
ANS=. ANS,<TMP
 end.
EIGEN,:}. ANS
)

char_lf_evec_sub=: 3 : 0
NB. modifyed char_lf
TR_SUM=. +/ tr MAT=. y NB. sum of trace
ANS=. <UMAT=. =/~ i. # y
for_LF. i.<: # y do.
MAT=. y +/ . * TMP=. MAT - UMAT *  TR_SUM
TR_SUM=. (% 2+LF)* +/  tr MAT
ANS=. ANS,<TMP
end.
)
```

```
norm0=: ] % [:%: [: +/ *:

pick_evec=: 3 : 0
TMP=.,./}. char_evec y NB. jisuuno narashi
|: ;("1) norm0 L:0 ,.({@> i. # TMP){"1 L:0 TMP
)
```

1. `a1=: 2 7 6,9 5 1,:4 3 8` NB. 最後の行はラミネート (,:) で

```
  a1
2 7 6
9 5 1
4 3 8
```

2. LF 法で固有値と特性方程式を求める (最初の 1 はカウンタの回数)

```
char_lf a1
-------------------------------------------
| 1 | 15 4.89898 _4.89898 | 360 _24 _15 1 |
-------------------------------------------
```

3. 多項式を解くと固有値が得られる。　p. は J の組み込みの多項式関数

$$360 - 24x - 15x^2 + x^3 = 0$$

```
  p. 360 _24 _15 1
----------------------------
| 1 | 15 4.89898 _4.89898 |
----------------------------
```

4. 固有ベクトルを求める

```
pick_evec a1
0.57735 _0.074915 _0.741582
```

```
0.57735 _0.666667  0.666667
0.57735  0.741582  0.074915
```

References

マイケル・ブラッドリー　松浦俊輔訳「数学を切りひらいた人びと　5」青土社　2009
W. ダンハム　黒川　若山　百々谷訳「オイラー入門」シュプリンガー東京　2004

J 言語は www.jsoftware.com から DL できます。QT 版と HTML 版があります。QT 版が動けば QT 版の方がお勧めです

DL してインストールしたら、Tool/Packages で全てのパッケージをインストールすると便利です

http://japla.sakura.ne.jp に多くの J 言語や APL の資料があります