

N桁区切り数系の 剰余算

Residue on the N - figure numbers system

山下氏の手計算への応援

中野嘉弘 (札幌市南区・85才)

NAKANO Yoshihiro (Sapporo, Japan)

FAX 011-588-3354 yoshihiro@river.ocn.ne.jp

多倍長演算が出来ない場合、数字データを単精度演算可能な5桁とか6桁づつに区切って、少しづつ遂行する方法がある。

そのような場合に有効と思われる方法を作成したので、取りあえず、報告する。諸賢の御改良を期待する。

0. は し が き

JAPLA 7月例会で中野は「(続)超長大数の分割 PARTITION 計算」を報告した。(文献1)。これは与数のサイズ14万以上の大数の分割であるから、未だ見なかった例である。これまで見られた最大値は3万程度であったから、これは大変、その結果要旨を英国コンピュータ学会の VECTOR 誌に報告・投稿した(文献2)。我が老友・山下氏からは早速、祝福の FAX も来た(文献3)。

こうした巨大数の PARTITION の個数の計算の正否は、対比すべき既知データが無いので、ラマヌジャン、アトキンその他の「相合式整除判定 Ramanujan や Atkin 法」で間接的に、云々する以外に方法は見当たらない。(もし、ご存じならば、請う御教示を!)ところが、次の FAX では、山下流「手計算」と中野流「整除判定」とでは結果が異なると来信があった(文献4)。これは大変! どちらかが間違っている? (文献4a) 畏友・西川会長は、それ以前から、PARTITION 計算に関して、山下「手計算」と同様な事を、コンピュータでやってみたらと、提案して居られた(文献5)。

そこで、中野は今回、その「実践」の試みを報告する。

1. 「手計算」での問題

山下流<手計算>の相合条件での話題は続く。(文献6)

- 1) pn86752 5, 11 で整除。 中野報告と同じ (OK)。
- 2) pn104055 11 で整除 (OK)、13 では (NG)、
中野報告は 13 でも整除 (OK)。
- 3) pn140005 末尾桁の 6 (中野値) を 7 に変更すれば 7 と 13 で整除可能。
中野報告では末尾桁が (原値) 6 のままで、7 と 13 で整除可、
問題は無し。
何故、このように異なるのか? <手計算>に原因があるか?

ロートル同士で真剣に突き合わせの結果、食い違いの原因はどうも、〈手計算〉におけるデータ値の〈転記ミス〉だと判明した。不幸な転記ミスを免れた例1)では問題なし。例2)では、山下側に3ヶ所のミスが判明した。例3)でも同様。と云う「転記ミス」が原因で一件落着!

では、何故、〈転記ミス〉を起こしたか?

中野は全て、コンピュータによる計算結果の一連の数値を、見易くする為、または〈手計算〉派の人々の為に、5桁ずつに区切って、報告した。ところが、山下氏の〈手計算〉は5桁ではなくて、6桁区切りに統一してあるとの事(文献6)。5桁データを、手動で6桁データに書き換えるのだから、作業は難物であると思われる。〈ミス〉の混入も無理も無いか!と、御同情申し上げる。

中野はそこで、〈手計算〉における〈区切りの桁数〉を、6桁などに限定せずに、可変に指定する事から始まって、幾つかの改良を試みた。以下はその報告である。山下流の計算は、「話のキッカケ」であって、それ自身を議論して居る訳では無い。内容的には独立のつもりである。本稿の話が有用ならば幸いである。

以下の数節は、説明用に、割合簡単な例を採る。

2. n-区切り関数 kugir 等

説明用に、直解しやすい例題として、整数 999 の PARTITION の数 $pn999 =: 23127843459154899464880444632250x$ (数字 32桁) を選ぼう。末尾の文字 x は、多精度整数 Extended precision integer constants の指定、即ち データタイプ関数

`dtype =: 3!:0` を apply すれば、結果値 8 を返すものである。

● 5桁ずつに区切る例

`k5g999 =. 5 kugiri pn999` kugiri 関数の方は途中経過も印刷する

dyadic な区切り関数 kugiri の左引数が区切り桁数、右引数が対象データ。
末尾詰め: 先頭組のみは5桁以下でもよいが、その他は5桁の記入を要する。
注意: もしも、末尾の組を3と記入したら30000と判断され、与データは2や5等で整除されることとなる。これらは明かに誤りである。以下同様。
演算結果 (関数 kugiri では、途中経過も印刷する。)

```
0 スタート
23
1 以上は 1 区切り。
23 12784
2 以上は 2 区切り。
23 12784 34591
3 以上は 3 区切り。
23 12784 34591 54899
4 同 4 区切り。
23 12784 34591 54899 46488
5 同 5 区切り。
23 12784 34591 54899 46488 04446
6 同 6 区切り。
23 12784 34591 54899 46488 04446 32250
7 同 7 区切り (完成)。
```

結果の検討

```
dtype k5g999 -> 2 character (文字型)
0 { k5g999 -> 2 先頭の文字
1 + 0 { k5g999 -> domain error (文字型との加算の故に)
1 + ". 0 { k5g999 -> 3 (数値型に変換して加算すれば OK)

$ k5g999 -> 37 文字数 (1区切り毎に末尾にblankが付くので)
_1{. k5g999 -> 最後の桁は blank
```

● 6桁ずつに区切る例 (山下流はこれのみ)

k6g999 =. 6 kugir pn999 ここに 区切り関数 kugir の左引数が区切り桁数、
kugir 関数 は途中は省略、結果のみを表示する。

結果

```
23 127843 459154 899464 880444 632250
```

3. 剰余関数

元来、山下氏の<手計算>とは、単精度 (6桁) の範囲で区切られた数値の剰余を
求める事であった。西川の示唆 (文献5) により、それに相当する事をコンピュータ
でやってみよう。

その為の関数を lres2c とする。 (同様の類似関数多し、Scripts 末尾)
被除数 k5g999は、前例の 多精度変数 pn999 に対応する文字型の区切られた変数とする。

| 除数 | 剰余関数 | 被除数 | 剰余 |
|-------------|---------------|-----------|--------------------------|
| 5 | lres2c k5g999 | 0 | (これは自明、データ末尾 0) |
| 3 | // | 0 | |
| 2 | // | 0 | |
| 7 | // | 0 | |
| 11 | // | 6 | (整除されぬ、50 - 11 x 4 = 6) |
| 13 | // | 0 | |
| 139 | // | 0 | |
| 281 | // | 0 | |
| 367 | // | 0 | |
| 47660563 | // | 0 | |
| 23619289868 | // | 6.55818e8 | (整除されぬ) |
| 23619289867 | // | 0 | (下行の検算参照) |

検算： PARTITION 999 の数値 pn999 (3 2桁) の既知の値の因数分解は
2*3^2*5^3*7^2*13*139*281*367*47660563*23619289867

4. 逆問題： 区切りデータの 復元一本化

説明の簡単例： 整数 70 の PARTITION pn70=: 4087968x (7桁)

これを3桁区切りしたもの

```
k3g70 =: 3 kugir pn70 -> 4 087 968
blankを探す '' = k3g70 -> 0 1 0 0 0 1 0 0 0 1
論理否定 -. '' = k3g70 -> 1 0 1 1 1 0 1 1 1 0
逆 invk3g70 =. (-. '' = k3g70) # k3g70 -> 4087968
```

データタイプ 逆 dtype invk3g70 -> 2 (文字型)
原 dtype pn70 -> 64 (拡張精度)

整合化 dtype ".invk3g70,'x'" -> 64 (拡張精度)

※注意: 記号 x を欠けば、値は 4.08796e6 タイプ 8 となるかも知れぬ。
その限界は9桁。従って当然、区切り桁数 < 9 を要する。
これらを実践する関数 invkugi, imvkugx は末尾に。

※参考 データタイプ

| | | | |
|------------------------|----|-----------------|-----|
| logical (論理型) | 0 | character (文字型) | 2 |
| integer (単精度整数) | 4 | floating (小数型) | 8 |
| complex (複素数型) | 16 | boxed (箱型) | 32 |
| x (Extended precision) | 64 | x: (rational) | 128 |

5. 諸関数の連動例

前例のデータ pn999 (3 2桁) を用いて説明する。

(7,5) lres2k pn999 5桁ずつに区切り、7で割った剰余を求める。
23 12784 34591 54899 46488 4446 32250 5桁区切り文字型データ
7 7組に区切られている
5 区切り桁数は5
0 剰余、今は0(7で整除)

(13,5) lres2k pn999
23 12784 34591 54899 46488 4446 32250 文字型データ
7 7組
5 5桁区切り
0 剰余、(13で整除)

(281,5) lres2k pn999
23 12784 34591 54899 46488 4446 32250 文字型データ
7 7組
5 5桁
0 281で整除

(7,6) lres2k pn999
23 127843 459154 899464 880444 632250 6桁区切り文字型データ
6 6組
6 6桁
0 7で整除

(13,6) lres2k pn999
23 127843 459154 899464 880444 632250
6 6組
6 6桁
0 13で整除

(5,4) lres2k pn999
2312 7843 4591 5489 9464 8804 4463 2250 4桁区切り文字型データ
8 8組

4 4桁
 0 5で整除
 (11,6) lres2k pn999
 23 127843 459154 899464 880444 632250
 6 6組
 6 6桁
 6 11 で剰余は 6

(5,7) lres2k pn999
 2312 7843459 1548994 6488044 4632250 7桁区切り
 5 5組
 7 7桁
 0 5 で整除

(3,8) lres2k pn999
 23127843 45915489 94648804 44632250 8桁区切り
 4 4組
 8 8桁
 0 3で整除

(13,8) lres2k pn999
 23127843 45915489 94648804 44632250
 4
 8
 0 13で整除

(139,8) lres2k pn999
 23127843 45915489 94648804 44632250
 4
 8
 0 139で整除

(138,8) lres2k pn999
 23127843 45915489 94648804 44632250
 4
 8
 114 138で割った時の剰余

6. 本番的巨大整数の分割データから

J 言語で計算中、A4 版の用紙に巨大整数を幾桁まで印刷出来ようか？
 筆者の経験では、横長の向きで、1 行 106 数字であり、PARTITION 問題で、被分割数
 が例えば 4 万台の時の、結果分割数 pn40003 では 2 行余 (218 数字) であった。
 余り長大であると、印字は途中から省略されてドット列になって仕舞いかねない。
 中野が問題にしている 被分割データが長大 1 4 万台、結果分割数 pn140005 が 411 桁
 の場合等には、それらのデータを移動するには、当然、何等かの注意・工夫が要る。
 例えば

1) データ型整数で扱うならば、4 回、行 copy し、末尾にキチンと、拡張精度の
 記号 x を付けねばならぬ。さらに appennd して、しめて 411 桁。
 しかし、パソコンで数表示を誤り無く copy 入力し、受取側が正しく再現出来るように
 丁寧にやらねばならぬ。やれば出来る等々は、しかし、数学や論理の話では無い。
 もしも手動でやったら、実際問題として、これは大変な作業である。

元来、拡張精度が可能ならば、あえて、手動の分割処理など、不必要であろう。
この方法は意味が無い！？

2) 拡張精度が不可能ならば、データの移動は、文字型として、小分けに分割して受け渡し、必要時に、小分けされた整数型に戻して計算せねばならぬ。
ただし、転記ミス、ブランクの正しい挿入に大変注意が必要であろう！
原データが、例えば5桁区切りであるのを、何らかの理由で、受取側が例えば6桁に直して転記しようとしたら、多少のミスは先ず、避けられまい！
以下、後者 2) の件を、本稿の方法で処理した実例を示す。

文字型5桁区切りデータ pb145 の作成

```
pb145a='6 98273 59954 '  
pb145b='93412 00512 76128 39473 61926 47824 60602 39116 74474 46296 '  
pb145c='51037 21875 67246 22592 56418 58387 80277 25283 07235 74229 '  
pb145d='34167 17184 99082 70376 96994 03562 07847 75383 76766 69051 '  
pb145e='14587 48814 30662 43670 88654 42756 02844 36481 85264 30676 '  
pb145f='00827 06824 25165 68898 24234 01556 59297 23605 62228 37855 '  
pb145g='61988 67304 61315 25344 91808 48182 04790 96545 22610 71534 '  
pb145h='14744 33769 76318 41392 81192 06098 77222 79671 07593 99381 '  
pb145i='95218 69188 58621 06276 88279 52648 35729 43187 61554 15706 '  
    appenndして纏める。  
pb145=: pb145a,pb145b,pb145c,pb145d,pb145e,pb145f,pb145g1,pb145h,pb145i  
dtype pb145
```

2 データ型は文字型

pb145

494 ブランクを含めた文字数

● 剰余計算例

```
7 lres2c pb145      7で pb145 を整除判定では剰余は 0  
83                  83組  
5                    5桁区切り  
0                    7で 整除
```

```
13 lres2c pb145     13で整除  
83  
5  
0                    整除
```

```
5 lres2c pb145      5では どうか？  
83  
5  
1                    剰余 1
```

```
11 lres2c pb145     11でも 剰余 1  
83  
5  
1
```

```
3 lres2c pb145      3では 剰余 2  
83  
5  
2
```

| | |
|--|---------------------|
| 2 lres2c pb145 83 5 0 | 2 では 整除される。 |
| NB. 7 lres2c pb145 83 5 0 | 7 で pb145 を割れば剰余は 0 |
| 13 lres2c pb145 83 5 0 | 13 で整除 |
| 5 lres2c pb145 83 5 1 | 5 では 剰余 1 |
| 11 lres2c pb145 83 5 1 | 11 でも 剰余 1 |
| 3 lres2c pb145 83 5 2 | 3 では 剰余 2 |
| 2 lres2c pb145 83 5 0 | 2 では 整除 |
| _1 { pb145 | pb145 の末尾はブランク |
| 5 { pb145 6 982 | pb145 の先頭の 5 文字 |
| 10 { pb145 6 98273 59 | 同上 10 文字 |
| _10 { pb145 554 15706 | 同上 末尾 10 文字 |
| NB. 17 lres2c pb145 83 5 6 | 17 で割れば剰余 6 |

NB.

ipb145=. invkugi pb145 5桁区切りの pb145 から ブランクを消去すれば
6 98273 59954 93412 00512 76128 39473 61926 47824 60602 39116 74474 46296 51037 21875
67246 22592 56418 58387 80277 25283 07235 74229 34167 17184 99082 70376 96994 03562
07847 75383 76766 69051 14587 48814 30662 43670 88654 42756 02844 36481 85264 30676 00...
6982735995493412005127612839473619264782460602391167447446296510372187567246225925
6418583878027725283072357422934167171849908270376969940356207847753837676669051145
8748814306624367088654427560284436481852643067600827068242516568898242340155659297
2360562228...

ipb145 結果は 411 文字 に戻る

411

NB.

ixpb145=. invkugx pb145 5桁区切り文字型の pb145 を 拡張精度整数に戻す
6 98273 59954 93412 00512 76128 39473 61926 47824 60602 39116 74474 46296 51037 21875
67246 22592 56418 58387 80277 25283 07235 74229 34167 17184 99082 70376 96994 03562
07847 75383 76766 69051 14587 48814 30662 43670 88654 42756 02844 36481 85264 30676 00...
6982735995493412005127612839473619264782460602391167447446296510372187567246225925
6418583878027725283072357422934167171849908270376969940356207847753837676669051145
8748814306624367088654427560284436481852643067600827068242516568898242340155659297
2360562228...
6982735995493412005127612839473619264782460602391167447446296510372187567246225925
6418583878027725283072357422934167171849908270376969940356207847753837676669051145
8748814306624367088654427560284436481852643067600827068242516568898242340155659297
2360562228...

ixpb145 成分は 1 (一続きのデータ)

1

7 | ixpb145 整除判定 by 7

0

13 | ixpb145 by 13

0

2 | ixpb145 by 2

0

5 | ixpb145 非整除 by 5

1

by 11

11 | ixpb145

1

10 {.: ixpb145 先頭の 10 数字を選べば

6982735995

dtype 10 {.: ixpb145 今のそれは文字型

2

_10 {.: ixpb145 同 末尾の 10 数字を選べば

6155415706

dtype _10 {.: ixpb145

2

dtype 10 { ixpb145 本来は拡張精度整数型

64

dtype _10 { ixpb145 末尾でも同様

64

NB.


```

pb145_6=. 6 kugir ixpb145   原データを 6 桁区切りにすれば
$ pb145_6
480
dtype pb145_6
2
20[. pb145_6
698 273599 549341 20
  _20 [. pb145_6
29431 876155 415706
  _21 [. pb145_6
729431 876155 415706
  24[. pb145_6
698 273599 549341 200512

```

- データの桁区切り等の各種変換操作と整除判定を、一つの関数に纏めることも可能。上例の幾つかには、既に利用してある。

6. ま と め

元来、J 言語の新版（拡張精度可能なもの）ならば不要な操作であろうが、旧版愛好のロートル仲間の誼みと西川会長のお勧めもあって、トライした小さな報告である。お役に立てば幸いである。

文 献

- 1) 中野嘉弘：「超長大整数の分割計算（第6報） 1 4万台 140,005 までの PARTITION」 JAPLA 2008/July/26 pp.10
- 2) NAKANO Yoshihiro " PARTITION HUGE INTEGERS Upto 140,005 VECTOR BCS UK (submitted 17 July 2008)
- 3) 山下紀幸： FAX(2008/7/27/8:49) 「月例資料多謝。 世界最高峰の pn140005 とはすごいですね。 せめて検算（7による相合関係だけでも確かめ）したいと思いますが・・・・・・」
- 4) 山下紀幸： FAX('08/ 7/27/ 15:57) 「超長大整数の分割計算中の pn140005 について、 第1桁目は 6 ではなくて 1 ではないでしょうか？ 1だと 7で整除されますが？」
 - － a) 中野返信 FAX(H20.7.27 Sun. pm.5:20 「中野報告（第6報 p.5）では、原値のまま、7と13で整除される。3,5,11では整除されない」とまで、既に書いてあります。 山下 FAXの意味不明！？」
- 5) 西川利男： FAX('08/5/30/pm.7:19) 「問題は数学かコンピュータか？ Jの多桁整数演算に限界があるのか？ 山下流<手計算>を、<手でなくコンピュータで行っては如何？>
- 6) 山下紀幸： FAX('08/ 7/29/ 12:00) 「7月資料（続）超長大整数の分割計算中の例 pn86752, pn104055, pn140005 と相合条件」

スク립ツ 類

```

kugir=: 3 : 0
:
bk=. ''
yc=. " : y
yn=. # yc
sen=. x | yn
sen1=. sen { . yc
ato=. yn - sen
NB. wr q: ato
nc=. ato % x
i=.0
ka=. (sen { . yc), bk
yc =. sen}. yc
i=.1
while. i < nc do.

ka=. ka, (x { . yc) , bk
yc =. x }. yc
i=.i+1
end.
ka
)

```

```

NB.
lres2k =: 3 : 0
:
xf=. 0{ x
xk=. 1{ x
wr Y=. " . yk=. xk kugir y
wr n=. # , Y
wr nye=. # " : _1 { . Y
i =. 0
xr =. xf | i { Y
while. i < n - 1 do.
i=.i+1
xr =. xf | (xr*(10^nye)) + i { Y
end.
xr
)

```

NB. kugir -> renzoku inverse transfm.

```

invkugi =: 3 : 0
NB. y character dtype 2, eliminate blank

wr Y =. y
wr (-.' '= Y) # Y
)

invkugx =: 3 : 0
wr " . (invkugi y),x'
)

```