数理解析研究所講究録
920巻 1995年 173-179

19.

An Introduction to MAGMA

脇 克志 (弘前大理)

19.1 Magmaとは

Magmaは、シドニー大学のJ.Cannonを中心とするComputational Algebra Groupにより主に代数学、数論、幾何学、組合せ論に関する問題を解く事を目的として開発された計算システムである。あつかう対象は有限構造(列、Graph、組)、演算の定義された有限集合(体、環、群、etc.)、基本的な無限集合(整数環、有理数体、複素数体)、またはこれらを組み合わせたものやその部分集合などである。Cannonによると“Magma”という言葉の由来は、Bourbaki[3]の第1章で2項演算の与えられた集合を“Magma”と呼ぶところから来ているそうである。ここでは、まぎらわしさをなくすため2項演算を持つ集合を代数構造と呼ぶ事とする。このComputational Algebra Groupは以前にCayleyと呼ばれる有限群とその表現を計算するシステムを開発しており、MagmaはCayleyの計算アルゴリズムおよび知識データベースを利用して作られている。また、M.Phostたちが開発した数論の問題を解くための計算システムKANTやH.Cohenが開発したPariシステムの一部もMagmaに組み込まれている。1993年8月23日から27日までロンドンでMagmaに関する国際シンポジウムが開催され、J.Cannon, G.Havas, E.O'Brien, D.Holt, M.Pohst, S.Linton, W.Bosma, G.Schneiderといった人達による講演が行われた。

ここではMagma V1.01-5を基にして、次の章でMagmaの他の数式処理システムにない特徴を述べ、3章で基本的な代数構造をMagmaではどのように扱っているかを紹介する。4章では、Magmaの有限群のデータベースの内容および利用方法を示し、最後にMagmaの具体的な利用例としてLintonのVector Enumerationアルゴリズムによる有限表現環の行列表現の構成とイデアルのメンバーシップ問題への応用を紹介する。

19.2 Magmaの特徴

Magma が Mathematica や Reduce といった数式処理システムと異なる点は主な計算対象を 2 項演算の定義された集合 (代数構造) とし、計算目標をその代数的構造を調べる事に置いている点である。また、数や数式の計算を行う場合でも、これらは必ず特定の代数構造の元とみなして計算が行われる。たとえば、数 1 が整数環に属している場合は $1+1=2$ となり、2 元体 GF(2) に属しているとすれば $1+1=0$ であり、有限群の単位元とみれば加法演算が定義されていないのでエラーメッセージが返ってくる事になる。Magma では各元に対しそれを含む代数構造をその元の Parent と呼んでいる。

このため、Magma にはあらかじめいろいろな代数構造が用意されており、利用者はこれらの代数構造を組み合わせる事により自分の研究対象を構成し、調べていくことになる。Magma で用意している代数構造としては、有限体、有理数体、実数体、行列環、整数環、それぞれの体および環上の多項式環、群環そして有限群などがある。特に、有限群に関しては Cayley で作られていた有限群のライブラリーをデータベースとして整えて使いやすくしている。

19.3 基本的な代数構造

この章では集合の定義と演算を紹介し、それから代表的な代数構造の定義方法を示していく。

19.3.1 有限集合の定義

最も単純な集合の定義方法はすべての元を { と } の間に並べる方法で例えば 1 から 19 までの奇数全体の集合を odds と定義する場合は、"odds:={ 1, 3, 5, 7, 9, 11, 13, 15, 17, 19 }" とすればよい。また、ある一定間隔で並んでいるものであれば、"odds:={ 1 .. 19 by 2};" と最大値と最小値および間隔を指定することにより定義することも出来る。さらに、例えば 1 から 10 までの巾乗数の集合 powers は "powers:={ x^2 : x in { 1 .. 10 } };" と変数とその定義域を使って定義することもできる。そして、もっとも一般的な定義はこの変数に対して条件をつけて集合を定義する方法である。すなわち、足して 15 となる 3 つの整数の組といった複雑な集合に対しても、"adds15:={ {a,b,c} : a,b,c in { 0 .. 15 } | a+b+c eq 15 };" と定義することが出来る。以上より Magma では有限集合を " { 集合の元 : 範囲 | 条件 }; " という形でかなり普通の数学で使われる集合の定義に近い形を利用できることが分かる。

19.3.2 集合の演算

いったん定義された集合に対してその和集合 (join)、共通部分 (meet)、差集合 (diff) といった集合演算ができる。また、"#集合名" でその集合の位数を表し、"&+集合名" でその集合の元全部の和、"&*集合名" でその集合の元全部の積を表す。例えば、

```
> odds:={2*x-1: x in {1..5}};
> print #odds;
5
> print &+odds;
25
> print &*odds;
945
```

となる。

19.3.3 群の定義

■有限表現群

一般的な群の定義の方法の 1 つとして生成元と関係式によるものがある。Magma では生成元も関係式も有限個である場合その群を有限表現群と呼んでいる。例えば、2 元 x、yで生成される関係式が $x^4 = y^2 = xyxy = e$ (e は単位元) という群であれば、"G:=Group<x,y|x^4=y^2=x*y*x*y=1>;" で定義できる。また、関係式の一部を変数にして系列的な群の定義も可能である。

```
> D := func<  n  |  Group< x, y | x^n=y^2=x*y*x*y=1 >  >;
> print Order(D(1));
2
> print Order(D(2));
4
> print Order(D(3));
8
```

上の例は群 D を 二面体群 D_n として定義したものである。

■可解群

生成元と関係式で与えられた群は、その計算に長い時間とたくさんのメモリーを必要とする。しかし次のような形で群を定義することにより、効率の良い計算が可能となる。

$$< a_1, \ldots, a_n | a_j^{p_j} = w_{jj}, (1 \leq j \leq n), a_j^{a_i} = w_{ij}, (1 \leq i < j \leq n) >;$$

ただし (1) p_jは $a_j^{p_j} \in < a_{j+1}, \ldots, a_n >$となる最小の素数 ($j < n$) で p_nは $a_n^{p_n}$が単位元となる最小の素数。(2) w_{ij} は $a_{i+1}, \ldots a_n$ でつくられた語 (word)。このような群の定義を power-conjugate presentation (pc-presentation) と呼ぶ。そして群が可解の場合、必ずこのような形で定義する事がで

きる。この pc-presentation がいくつかの追加条件を満たす場合その群の任意の元は

$$a_1^{\alpha_1} \cdots a_n^{\alpha_n}, (0 \leq \alpha_i < p_i, \text{ for } i = 1, \ldots, n)$$

と一意的に表される (これを標準形と呼ぶ)。Magma では、p-群および可解な置換群に対してこの pc-presentation を与える関数が用意されているが (pQuotient とPCGroup)、一般に可解な有限表現群の pc-presentation を与えるアルゴリズムはまだ知られていない。しかし代表的な可解群については、関数として用意されている。例えば、二面体群や巡回群であれば

```
> E:=DihedralGroup(GrpPC,5);
> A:=CyclicGroup(GrpPC,8);
> print E;
GrpPC : E of order 10 = 2 * 5
PC-Relations:
E.1^2 = Id(E),
E.2^5 = Id(E),
E.2^E.1 = E.2^4
> print A;
GrpPC : A of order 8 = 2^3
PC-Relations:
A.1^2 = A.2,
A.2^2 = A.3
```

と定義できる。(注意：群 A について 3 番目の生成元に関する関係式が表示されていないが内部ではきちんと定義されている。)

■置換群

置換群は、ある対称群の部分群として定義される。よって各元は特定の対称群に必ず属し、同じ表現の元 (1,2) でも 2 次の対称群の元と 6 次の対称群の元は明確に区別される。よって置換群の定義は、その対称群の次数と生成元によって定義される。例えば、元 (1,2) と (1,2,4) で生成される 4 次の対称群の部分群は "PermutationGroup<4|(1,2),(1,2,4)>;" と定義される。また集合の定義方法を利用すると、

```
> G:=SymmetricGroup(7);
> S:=[  G   |   (i,i+2) : i in [1..5 by 2]   ];
> H:=sub<G|S>;
> print H;
Permutation group H acting on a set of cardinality 7
    (1, 3)
    (3, 5)
    (5, 7)
```

といった定義も可能である。

■行列群

行列群は、ある体またはユークリッド整域 R上の一般線形群 GL(n,R) の部分群として定義される。例えば、2 元体 GF(2) 上で元 $\begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}$ と $\begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$ で生成される群は

“ MatrixGroup< 3, GF(2) | [0,0,1,1,0,0,0,1,0], [0,1,0,1,0,0,0,0,1] >; ”
で定義される。

19.3.4 環の定義

■整数環

整数環は、“Integers();” で定義される。次の例は整数環を使って 2 および 3 の倍数の無限集合を定義したものである。

```
> MultOf2:={! n in Integers() | n mod 2 eq 0 !};
> print 4 in MultOf2;
true
> MultOf3:={! n in Integers() | n mod 3 eq 0 !};
> print 4 in (MultOf2 meet MultOf3);
false
> print 6 in (MultOf2 meet MultOf3);
true
```

■多項式環

Magma では基本的に 1 変数の多項式環を扱う。しかし任意の環上の多項式環を定義できるので、多項式環上の多項式環を考えると多変数多項式環を定義することが可能となる。整数環上の変数 x、yで定義される 2 変数多項式環 S は “S<x,y>:=PolynomialRing(Integers(),2);” となる。しかしこの場合変数 x と yは対等ではない。実際 “print Coefficents(2*x*y+x+y);” とやると “[x,2*x+1]” と答える。これは、式 $2xy+x+y$ の係数は x と $2x+1$ であるということで S は x による 1 変数多項式環上の yによる 1 変数多項式環であることが分かる。Magma では、Mathematica や Maple などで利用可能な多項式に関する演算はまだ十分でそろっていない。特に Gröbner basis がまだ実装されていないことはたいへん残念である。以下に他の数式処理システムにはない多項式に置換群や行列群の元を作用させる機能の実行例を示す。

```
> S<x,y,z>:=PolynomialRing(GF(3),3);
> f:=x*y+y+z;
> print f;
z + (x + 1)*y
> G:=SymmetricGroup(3);
> a:=G ! (1,2,3);
> print a;
(1, 2, 3)
> print f^a;
(y + 1)*z + x
> M:=GeneralLinearGroup(3,GF(3));
> b:=M ! [1,1,0,1,0,1,0,1,1];
> print b;
[1 1 0]
[1 0 1]
[0 1 1]
> print f^b;
(y + (x + 2))*z + (x + 1)*y + x^2 + x
```

■有限表現環

有限表現環はある環上で有限個の生成元と関係式で定義される環である。この方法で非可換環の群環なども定義することができる。しかし現段階では、イデアルの定義はできても後で述べる Vector Enumeration 以外役に立ちそうな機能はまだない。

```
> k:=GF(3);
> M<x,y>:=Monoid<x,y|x*y=y*x>;
> F<x,y>:=FreeAlgebra(k,M);
> I:=rideal<F|x^2*y^3-1,x*y^6-1>;
> print x^2*y^3-1 in I;
false
```

上の例では 3 元体 GF(3) 上の 2 変数多項式環 F を作り、この F の元 x^2y^3-1 と xy^6-1 で生成されるイデアルを I としているがメンバーシップ問題はうまくいかないようである。

19.4 有限群のデータベース

現在 Magma のライブラリの中の有限群のデータベースは全部で 12 個ある。各データベースは次のような名前を持っている。

glnzgps 整数環上の n 次一般線形群 $(n = 2, 3, 4, 5)$ の極大有限部分群。

gps100 位数が 100 以下の非可換可解群。(ただし位数が 64 のものは除く)

isolgps 有限体上の一般線形群 GL(n,p) の既約可解部分群。(ただし $p^n < 255$)

matgps 有限体上の行列群。

perfgps 位数が 100 万以下のすべての完全群。

pergps 置換群。

prmgps 次数が 50 以下の原始的な群。

simgps 位数が 100 万以下のすべての単純群。

solgps pc-presentation で定義された 13 の可解群。

thrgps 位数が 729 以下の 3-群。

trmgps 次数が 2 から 12 の推移な置換群。

twogps 位数が 256 以下の 2-群。

各データベースは "load データベース名;" で Magma に読み込まれる。利用方法は、"?データベース名;" で知ることが出来る。例えば、データベース "gps100" を使って位数が 30 から 40 の間の群で共役類の個数が 6 の群を探すのであれば次のようにすればよい。

```
> load gps100;
Loading "/usr3/usr32/magma-V1.0.1-5/LIBS/gps100/gps100"
> P := GroupOfOrderProcess(<30, 40>);
> while not GroupProcessIsEmpty(P) do
while>        G := GroupProcessExtractGroup(P);
while>        if NumberOfClasses(G) eq 6 then
while|if>           print G;
while|if>     end if;
while>        GroupProcessNext(~P);
while> end while;
GrpPC : G of order 36 = 2^2 * 3^2
PC-Relations:
G.1^2 = G.2,
G.2^2 = Id(G),
G.3^3 = Id(G),
G.4^3 = Id(G),
G.3^G.1 = G.4^2,
G.3^G.2 = G.3^2,
G.4^G.1 = G.3,
G.4^G.2 = G.4^2
```

ここでは条件を満たす群が 1 つだけあり、その位数 36 であることが分かる。

19.5 利用例

非可換を含めて一般の有限表現環に対してそのイデアルのメンバーシップ問題は容易ではない。そこで、 Magma において有限表現環に与えられた唯 1 つの道具を使ってイデアルのメンバーシップ問題を解いてみたい。この章ではまず Vector Enumeration アルゴリズムを説明し、次にそれを使ってイデアルのメンバーシップ問題を解く方法を示す。

19.5.1 Linton の Vector Enumeration

ある体 k 上の有限表現環 R とそのイデアル I が与えられ R/I が k上有限次元 (n) であると仮定する。このとき R の I に関する剰余類 $\{r_1+I,\ldots,r_n+I\}$ を基底とする k 上のベクトル空間 V を考える。Linton の Vector Enumeration とは、この V を R-加群と見た時の R の生成元の V に対する作用の行列表現を与えるアルゴリズムである。このアルゴリズムの利点は R や I の大きさに関係なく R/Iの次元が十分小さい場合、計算が出来る点である。Magma では関数 "QuotientModule" でこの計算を行う。次の例は、有理数体上の 2 変数 x、yの多項式環 F を構成しその 2 つの元 x^2y^3-1 と xy^6-1 から生成されるイデアル I を定義する。この F と I から関数 "QuotientModule" は F の生成元 x と y の V に対する作用の行列表現 mats、F-加群 V の生成ベクトル im、V の基底となる Fの Iによる剰余類の代表系の列 preim を計算する。