モンゴメリ乗算

1.モンゴメリ写像

X＝{x|0≦x＜N，N は 2 以上の自然数}とし，R を N と互いに素な自然数として，次のような X から X への写像 f を考える。

$$y = f(x) = Rx \bmod N$$

このとき， $R^{-1}R \bmod N = 1$ となる $R^{-1} \in X$ が存在し，

$$f(a) = f(b) \Rightarrow Ra \bmod N = Rb \bmod N \Rightarrow R^{-1}Ra \bmod N = R^{-1}Rb \bmod N \Rightarrow a = b$$

となる。よって，

$$a = b \Leftrightarrow f(a) = f(b)$$

が成り立つから， f は 1 対 1 写像（単写）である。

f は有限集合 X から X への写像であるから上への写像（全射）にもなる。

ゆえに， f は X から X への 1 対 1 上への写像（全単射）であり， f^{-1} が存在する。

2.モンゴメリ乗算

$a,b \in X$ に対して、積 a×b は ab mod N を表すものとする。このとき

$$f(a \times b) = R \times a \times b$$
$$= R \times a \times R \times b \times R^{-1}$$
$$= f(a) \times f(b) \times R^{-1}$$

よって， $A = f(a), B = f(b)$ とすると

$$f(a \times b) = A \times B \times R^{-1}$$

ここで，演算 $\otimes$ を

$$A \otimes B = A \times B \times R^{-1}$$

と定める。このとき

$$f(a \times b) = A \otimes B = f(a) \otimes f(b) \cdots ①$$

演算 $\otimes$ をモンゴメリ乗算という。モンゴメリ乗算は，次の性質をもつ。

$$A \otimes B = B \otimes A \quad \text{（交換法則）}$$
$$(A \otimes B) \otimes C = A \otimes (B \otimes C) \quad \text{（結合法則）}$$

証明

交換法則

①より

$$A \otimes B = f(a) \otimes f(b) = f(a \times b) = f(b \times a) = f(b) \otimes f(a) = B \otimes A$$

結合法則

$C = f(c)$ とする。①より

$$(A \otimes B) \otimes C = \{f(a) \otimes f(b)\} \otimes f(c) = f(a \times b) \otimes f(c) = f((a \times b) \times c) = f(a \times (b \times c))$$
$$= f(a) \otimes f(b \times c) = f(a) \otimes \{f(b) \otimes f(c)\} = A \otimes (B \otimes C)$$

証明終

次のようにしても証明できる。

$$(A \otimes B) \otimes C = (A \times B \times R^{-1}) \times C \times R^{-1}$$
$$= A \times (B \times C \times R^{-1}) \times R^{-1}$$
$$= A \otimes (B \otimes C)$$

これからは $(A \otimes B) \otimes C$ ， $A \otimes (B \otimes C)$ を単に， $A \otimes B \otimes C$ と書く。

また，結合法則の証明から

$$f(a\times b\times c)=f(a)\otimes f(b)\otimes f(c)=A\otimes B\otimes C$$

は明らかである。

一般に，$A_k=f(a_k)\quad(k=1,2,3,\cdots,n)$ とすると，

$$f(a_1\times a_2\times\cdots\times a_n)=f(a_1)\otimes f(a_2)\otimes\cdots\otimes f(a_n)=A_1\otimes A_2\otimes\cdots\otimes A_n$$

が成り立つ。したがって

$$a_1\times a_2\times\cdots\times a_n=f^{-1}(f(a_1)\otimes f(a_2)\otimes\cdots\otimes f(a_n))=f^{-1}(A_1\otimes A_2\otimes\cdots\otimes A_n)$$

特に，累乗について

$$f(a^n)=f(a)\otimes f(a)\otimes\cdots\otimes f(a)=A^n$$　（右辺はモンゴメリ乗算による累乗）

より，　$a^n=f^{-1}(A^n)$

3.モンゴメリリダクション

$$A\otimes B=A\times B\times R^{-1}$$

において，$AB=T$ とおくと　（$T<N^2$）

$$A\otimes B=TR^{-1}\bmod N$$

すなわち，モンゴメリ乗算 1 回毎に $MR(T)=TR^{-1}\bmod N$ を計算する必要がある。$MR(T)$ を求める処理を，モンゴメリリダクションという。このとき

$$f^{-1}(A)=MR(A)$$

が成り立つ。ここで

$$R=2^m>N$$　（m は自然数）

としたとき，$MR(T)$ を高速に計算するアルゴリズムを示す。

1. 事前に　$N'N=-1\bmod R$　となる N' を求める。
2. $s=(T\bmod R)N'\bmod R$
3. $t=(T+sN)/R$
4. if　t>=N　then　t=t-N

このとき　$t=TR^{-1}\bmod N=MR(T)$　が成り立つ。

1.において

$$N'=N^{-1}(R-1)\bmod R$$

であり，N^{-1} は拡張ユークリッドの互除法により求められる。

また，

$$T\bmod R=T\ \&\ 0xFF\cdots F\qquad\text{ただし，}\ 0xFF\cdots F=2^m-1=R-1$$

$$(T+sN)/R=(T+sN)>>m$$

であるから，2～4 の計算は高速に行われる。

証明

まず，3 において，$T+sN$ が R で割り切れることを示す。

$$T+sN\equiv T+TN'N\equiv T-T\equiv 0\pmod R$$

ゆえに，$T+sN$ は R で割り切れる。

次に，　$t\equiv TR^{-1}\pmod N$　を示す。

$$t = (T + sN) / R$$

より $tR = T + sN$

よって $tR \equiv T \pmod{N}$

したがって $tRR^{-1} \equiv TR^{-1} \pmod{N}$

ゆえに $t \equiv TR^{-1} \pmod{N}$

最後に，$t < 2N$ を示す。

$T < N^2 < NR, s < R$ より $T + sN < NR + RN = 2NR$

よって $t = (T + sN) / R < 2N$

以上により，$t = TR^{-1} \bmod N = MR(T)$ が成り立つ。

4.モンゴメリリダクションの応用

モンゴメリリダクションを用いて $f(a)$ を計算する方法を示す。

$$S = R^2 \bmod N$$

とすると

$$\begin{aligned} A &= f(a) \\ &= Ra \bmod N \\ &= aR^2R^{-1} \bmod N \\ &= aSR^{-1} \bmod N \\ &= MR(aS) \end{aligned}$$

また，

$$a = f^{-1}(A) = MR(A)$$

であった。

モンゴメリリダクションを用いて $a \bmod N$ を計算する方法を示す。

$$\begin{aligned} a \bmod N &= aR^{-1}R^2R^{-1} \bmod N \\ &= \{(aR^{-1})S\}R^{-1} \bmod N \\ &= MR(MR(a)S) \end{aligned}$$

以下は，理論を確かめるためのサンプルプログラムです。

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

//s*a+t*b=c となる s,t,c を求める(拡張ユークリッドの互除法)
void ExGCD(long a,long b,long *s,long *t,long *c) {
    long r,u,v;
    if (b==0)
    {
        *s=1;
        *t=0;
        *c=a;
```

```
	}else{
		ExGCD(b,a%b,&u,&v,c);
		*s=v;
		*t=u-v*(a/b);
	}
}

unsigned long inverse(long a,long b) //逆元を求める
{
	long s,t,c;
	ExGCD(a,b,&s,&t,&c);
	t%=a;
	if (t<0) t=t+a;
	return (unsigned long)t;
}

#define N 5657
#define SHIFT 15
#define R (0x1<<SHIFT)
#define S ((R*R)%N)

unsigned long MR(unsigned long N_1,unsigned long T)
{
	unsigned long s,t;
	s=((T&(R-1))*N_1)&(R-1);
	t=(T+s*N)>>SHIFT;
	if (t>=N) t=t-N;
	return t;
}

int main(void)
{
	unsigned long N_1,a,b,c,d,e;
	unsigned long i;
	N_1=(inverse(R,N)*(R-1))&(R-1);
	printf("%d",N_1);
	printf("¥n");
	clock_t start, end;
	start = clock();
	for ( i = 0 ; i<100000000 ; i++){   //100000000 回の計算時間を測定して性能を調べる
		a=rand();
		b=MR(N_1,a);
```

```
        c=b*S;
        d=MR(N_1,c);
        e=a%N;
        if (d!=e) printf("異なる");
    }
    end = clock();
    printf("%f 秒　¥n", (double)(end - start) / CLOCKS_PER_SEC);
    printf("%d %d",d,e);
    printf("¥n");
    system("PAUSE");
    return 0;
}
```