情 報 解 析 学 特 講

── 暗 号 論 入 門 ──

高知大学大学院 理学研究科 情報科学専攻

塩 田 教 官

(平成 9 年度 1 学期)

目 次

1 初等整数論からの準備

1.1 ユークリッドのアルゴリズム

整数全体の集合を $\mathbf{Z}$ と表す。

1.1.1 最大公約数

ふたつの整数 $a, b \in \mathbf{Z}$ の最大公約数を (a, b) と表す。

1.1.2 互いに素

ふたつの整数 $a, b \in \mathbf{Z}$ は、その最大公約数 (a, b) が 1 であるとき **互いに素である** と言う。

1.1.3 補題

$b \neq 0$ のとき、a を b で割った余りを r とおくと $(a, b) = (b, r)$ が成り立つ。

1.1.4 定理

$(a, b) = d$ とおくとき、$d = ax + by$ を満たす整数 x, y が存在する。しかも、d, x, y は次のユークリッドのアルゴリズムによって (高速に) 計算することができる。

1.1.5 ユークリッドのアルゴリズム

(1) $b = 0$ ならば

$$d := |\, a \,|, \qquad x := \begin{cases} 1 & (\ a \geqq 0 \text{ のとき }) \\ -1 & (\ a < 0 \text{ のとき }) \end{cases}, \qquad y := 0$$

とせよ。

(2) $b \neq 0$ ならば

a) 次の様に数列 $\{\, r_n \,\}, \{\, x_n \,\}, \{\, y_n \,\}$ を作る：

$$r_0 := a, \quad r_1 := b, \quad x_0 := 1, \quad x_1 := 0, \quad y_0 := 0, \quad y_1 := 1,$$

$$\begin{cases} q \ := r_{n-2} \text{ を } r_{n-1} \text{ で割った商} \\ r_n := r_{n-2} - q \times r_{n-1} = r_{n-2} \text{ を } r_{n-1} \text{ で割った余り} \\ x_n := x_{n-2} - q \times x_{n-1} \\ y_n := y_{n-2} - q \times y_{n-1} \end{cases} \qquad (n = 2, 3, \cdots).$$

b) $r_n = 0$ となるまで数列を計算し、その時点の n で、

$$\begin{cases} d := r_{n-1}, & x := x_{n-1}, & y := y_{n-1} & (\ r_{n-1} > 0 \text{ のとき }) \\ d := -r_{n-1}, & x := -x_{n-1}, & y := -y_{n-1} & (\ r_{n-1} < 0 \text{ のとき }) \end{cases}$$

と置け。

1.1.6 証明

r_n の絶対値は減少列なので、有限回で $r_n = 0$ となる。このとき補題より

$$(a, b) = (r_0, r_1) = (r_1, r_2) = \cdots = (r_{n-1}, r_n) = |\, r_{n-1}\,|$$

となる。更に帰納法によって、各ステップにおいて

$$r_n = ax_n + by_n$$

が成り立つことがわかる。□

1.1.7 系

a と n が互いに素ならば $ax + ny = 1$ を満たす整数 x, y がユークリッドのアルゴリズムによって求まる。

1.2 群

1.2.1 群

次の条件を満たす集合 G を **群** と言う。

(1) G は演算 (以下積で書く) を持つ。

(2) 演算は結合律を満たす： $(ab)\,c = a\,(bc)$ for $\forall\, a, b, c \in G.$

(3) 演算の単位元 e が存在する： $ae = ea = a$ for $\forall\, a \in G.$

(4) $\forall\, a \in G$ は演算の逆元 a^{-1} を持つ： $\forall\, a \in G\,,\quad \exists\, a^{-1} \in G$ s.t. $aa^{-1} = a^{-1}a = e.$

1.2.2 アーベル群

演算が交換律を満たすとき、G を **アーベル群 (可換群)** と言う： $ab = ba$ for $\forall\, a, b \in G.$

1.2.3 有限群

G の要素が有限個のとき、G を **有限群** と言う。また要素の個数を G の **位数** と言い、$|G|$ と表す。

1.2.4 補題

G が群で $a, b, c \in G$ のとき、 $ab = ac$ または $ba = ca$ $\Longrightarrow$ $b = c.$

1.2.5 定理

有限群 G の 位数を N と置くとき、 $a^N = e$ for $\forall\, a \in G.$

1.2.6 元の位数

群 G の元 a について $a^n = e$ を満たす自然数が存在するとき、そのような最小の自然数を **元 a の位数** と呼ぶ。

1.2.7 補題

群 G の元 a と、ふたつの 0 でない整数 m, n について $a^m = a^n = e$ が成り立てば、m, n の最大公約数 $d = (m, n)$ に対して $a^d = e$.

1.2.8 証明

定理 **1.1.4** より $d = mx + ny$ となる整数 x, y が取れ、$a^d = a^{(mx+ny)} = (a^m)^x (a^n)^y = e$.

1.2.9 定理

(1) 群 G の元 a と整数 m に対して $a^m = e$ が成り立てば m は a の位数の倍数である。

(2) 有限群 G において、任意の元の位数は群の位数の約数である。

1.2.10 巡回群

群 G に　$G = \{\cdots, a^{-2}, a^{-1}, e, a, a^2, \cdots\}$　を満たす元 a が存在するとき、$G = \langle a \rangle$ と表して、G を a **によって生成される有限群** と言う。また a を **巡回群** G **の生成元** と言う。

1.3 法演算

1.3.1 法

2 以上の自然数 n を固定し、以下これを **法**（ほう）と呼ぶ。

1.3.2 合同式

ふたつの整数 a, b に対して、$a - b$ が n で割り切れるとき、a **と** b **は** n **を法として合同である** と言い、

$$a \equiv b \pmod{n}$$

と表す（a 合同 b モド n と読む）。法が明らかな場合には $(\bmod n)$ を省略しても良い。

1.3.3 剰余類

n による剰余系の集合を $\mathbf{Z}/n\mathbf{Z}$ と表す。また $a \in \mathbf{Z}$ を含む剰余類を $\bar{a}$ または $a \bmod n$ の様に表す。

$$\mathbf{Z}/n\mathbf{Z} = \{\bar{0}, \bar{1}, \cdots, \overline{n-1}\}$$

である。

1.3.4 剰余類の加法・乗法

$\mathbf{Z}/n\mathbf{Z}$ には次の方法で加法と乗法が定義できる：

$$\bar{a} + \bar{b} := \overline{a+b}, \qquad \bar{a}\bar{b} := \overline{ab}$$

1.3.5 定理

$\mathbf{Z}/n\mathbf{Z}$ は加法に関して $\bar{0}$ を単位元とする位数 n の有限アーベル群を成す。

1.3.6 既約剰余類

$\mathbf{Z}/n\mathbf{Z}$ の元のうち、n と互いに素な $a \in \mathbf{Z}$ を含む剰余類を **既約剰余類** と言う。既約剰余類全体の集合を $(\mathbf{Z}/n\mathbf{Z})^\times$（クロス）または $(\mathbf{Z}/n\mathbf{Z})^*$（スター）と表す。

1.3.7 定理

$(\mathbf{Z}/n\mathbf{Z})^\times$ は乗法に関して $\bar{1}$ を単位元とする有限アーベル群を成す。

1.3.8 証明

逆元の存在を示す。ユークリッドのアルゴリズムによって $ax + ny = 1$ を満たす整数 x, y が存在するので、

$$1 = ax + ny \equiv ax \quad (\bmod n)$$

従って $\bar{x}$ が $\bar{a}$ の逆元になる。□

1.3.9 オイラーの関数

$(\mathbf{Z}/n\mathbf{Z})^\times$ の位数、すなわち、0 から $n-1$ までの整数の中で法 n と互いに素である整数の個数を $\varphi(n)$ と表す。これを **オイラーの関数** と呼ぶ。

1.3.10 オイラーの定理

整数 a が法 n と互いに素であるとき、

$$a^{\varphi(n)} \equiv 1 \quad (\bmod n)$$

が成り立つ。

1.3.11 法が素数の場合

法が素数 p のとき、

$$(\mathbf{Z}/p\mathbf{Z})^\times = \{\, \bar{1}, \bar{2}, \cdots, \overline{p-1} \,\}, \qquad \varphi(p) = p - 1.$$

1.3.12 法が素数べきの場合

法が素数べき p^e のとき、

$$(\mathbf{Z}/p^e\mathbf{Z})^\times = \{\, \bar{x} \,|\, 0 \leqq x < p^e \text{ , } x \text{ は } p \text{ で割り切れない} \,\}, \qquad \varphi(p^e) = p^e - p^{e-1}.$$

1.3.13 フェルマーの小定理

法が素数 p のとき、$a \in \mathbf{Z}$ が p で割り切れなければ

$$a^{p-1} \equiv 1 \quad (\bmod p).$$

従って、任意の $a \in \mathbf{Z}$ に対して

$$a^p \equiv a \quad (\bmod p).$$

1.3.14 定理

(1) 法が素数 p のとき、$(\mathbf{Z}/p\mathbf{Z})^\times$ は乗法に関して位数 $p-1$ の巡回群 を成す。すなわち或る元 $g \in (\mathbf{Z}/p\mathbf{Z})^\times$ が存在して

$$(\mathbf{Z}/p\mathbf{Z})^\times = \langle\, g \,\rangle = \{\, 1 = g^0, g = g^1, g^2, \cdots, g^{p-2} \,\}$$

となる。このような g (巡回群の生成元) を mod p の原始根と呼ぶ。

(2) mod p の原始根は (mod p で) $\varphi(p-1)$ 個存在する。g をひとつの mod p の原始根とすると、

$$g^j \qquad (\, 1 \leqq j \leqq p-2, \quad (j, p-1) = 1 \,)$$

がすべての原始根を与える。

1.3.15 例

$$(\mathbf{Z}/3\mathbf{Z})^{\times} = \{\, 2^0 = 1,\ 2 \,\}$$

$$(\mathbf{Z}/5\mathbf{Z})^{\times} = \{\, 2^0 = 1,\ 2,\ 2^2 = 4,\ 2^3 = 3 \,\}$$

$$(\mathbf{Z}/7\mathbf{Z})^{\times} = \{\, 3^0 = 1,\ 3,\ 3^2 = 2,\ 3^3 = 6,\ 3^4 = 4,\ 3^5 = 5 \,\}$$

$$(\mathbf{Z}/11\mathbf{Z})^{\times} = \{\, 2^0 = 1,\ 2,\ 2^2 = 4,\ 2^3 = 8,\ 2^4 = 5,\ 2^5 = 10,\ 2^6 = 9,\ 2^7 = 7,\ 2^8 = 3,\ 2^9 = 6 \,\}$$

$$(\mathbf{Z}/13\mathbf{Z})^{\times} = \left\{ \begin{array}{c} 2^0 = 1,\ 2,\ 2^2 = 4,\ 2^3 = 8,\ 2^4 = 3,\ 2^5 = 6,\ 2^6 = 12,\ 2^7 = 11,\ 2^8 = 9,\ 2^9 = 5, \\ 2^{10} = 10,\ 2^{11} = 7 \end{array} \right\}$$

1.3.16 定理

法が奇素数のべき p^e のとき、$(\mathbf{Z}/p^e\mathbf{Z})^{\times}$ は乗法に関して位数 $p^{e-1}(p-1)$ の巡回群を成す。その巡回群としての生成元を $\bmod p^e$ の原始根と呼ぶ。

1.4 中国剰余定理

1.4.1 中国剰余定理（ 法がふたつの場合 ）

自然数 m, n が互いに素のとき、ふたつの整数 a, b に対して連立合同式

$$x \equiv a \quad (\bmod m), \qquad x \equiv b \quad (\bmod n)$$

は mn を法として唯ひとつの解 x を持つ。

1.4.2 中国剰余定理のアルゴリズム

ユークリッドのアルゴリズムを m と n に適用して

$$mu + nv = 1$$

を満たす整数 u, v を求め、

$$x := bmu + anv$$

と置け。

1.4.3 証明

m を法とすれば

$$x \equiv anv = a(1 - mu) \equiv a \qquad (\bmod m).$$

n を法としても同様。□

1.4.4 系

自然数 m, n が互いに素のとき、連立合同式

$$x \equiv a \quad (\bmod m), \qquad x \equiv a \quad (\bmod n)$$

の解は $x \equiv a \ (\bmod mn)$ である。

1.4.5 系

自然数 m, n が互いに素のとき

$$\varphi(mn) = \varphi(m)\,\varphi(n)$$

が成り立つ。

1.4.6 系

自然数 n の素因数分解を

$$n = p^e q^f \cdots r^g$$

とすれば

$$\varphi(n) = (p^e - p^{e-1})(q^f - q^{f-1}) \cdots (r^g - r^{g-1}) = n\left(1 - \frac{1}{p}\right)\left(1 - \frac{1}{q}\right)\cdots\left(1 - \frac{1}{r}\right)$$

1.4.7 中国剰余定理（ 一般の場合 ）

s 個の自然数 $m_1, m_2, \cdots, m_s$ がどの 2 つも互いに素のとき、s 個の整数 $a_1, a_2, \cdots, a_s$ に対して連立合同式

$$x \equiv a_1 \ (\bmod m_1\,), \quad x \equiv a_2 \ (\bmod m_2\,), \ \cdots\ , \ x \equiv a_s \ (\bmod m_s\,)$$

は $m_1 m_2 \cdots m_s$ を法として唯ひとつの解 x を持つ。

1.4.8 中国剰余定理（ 一般の場合 ）のアルゴリズム

解 x は次のアルゴリズムにより求まる：

(1) ユークリッドのアルゴリズムを m_1 と $M_1 = m_2 \cdots m_s$ に適用して

$$m_1 u_1 + M_1 v_1 = 1$$

を満たす整数 u_1, v_1 を求め、$w_1 := M_1 v_1$ とおく。このとき、

$$w_1 \equiv 1 \ (\bmod m_1\,), \quad w_1 \equiv 0 \ (\bmod m_2\,), \ \cdots\ , \ w_1 \equiv 0 \ (\bmod m_s\,)$$

が成り立つ。

(2) 同様にして各 j に対して

$$w_j \equiv 1 \quad (\bmod m_j\,), \quad w_j \equiv 0 \quad (\bmod m_k\,) \quad (k \neq j\,)$$

を満たす整数 w_j を求める。

(3) $x := a_1 w_1 + a_2 w_2 + \cdots + a_s w_s$ が解となる。

1.5 平方剰余記号・ヤコビ記号

1.5.1 平方剰余

奇素数 p を法として考える。p と互いに素な整数 a は、合同式

$$x^2 \equiv a \quad (\bmod p\,)$$

が解を持つとき **法 p に関する平方剰余である** と言い、そうでないとき **平方非剰余である** と言う。

1.5.2 平方剰余記号

平方剰余記号（ルジャンドル記号とも言う）とは、奇素数 p と整数 a に対して

$$\left(\frac{a}{p}\right) := \begin{cases} 1 & a \text{ が法 } p \text{ に関する平方剰余のとき} \\ -1 & a \text{ が法 } p \text{ に関する平方非剰余のとき} \\ 0 & a \text{ が } p \text{ で割り切れるとき} \end{cases}$$

で定められる。(分数と同じ記号だが、状況により区別して用いよ。)

1.5.3 例

$2 \equiv 9 = 3^2 \ (\bmod 7)$ なので $\left(\frac{2}{7}\right) = 1.$

また $x^2 \equiv 3 \ (\bmod 7)$ となる整数 x は存在しないので $\left(\frac{3}{7}\right) = -1.$

1.5.4 定理 (オイラーの規準)

任意の $a \in \mathbf{Z}$ に対して $\left(\frac{a}{p}\right) \equiv a^{\frac{p-1}{2}} \ (\bmod p)$ が成り立つ。

1.5.5 例

$\left(\frac{2}{7}\right) \equiv 2^{\frac{7-1}{2}} = 2^3 = 8 \equiv 1 \ (\bmod 7)$ なので $\left(\frac{2}{7}\right) = 1.$

また $\left(\frac{3}{7}\right) \equiv 3^{\frac{7-1}{2}} = 3^3 = 27 \equiv -1 \ (\bmod 7)$ なので $\left(\frac{3}{7}\right) = -1.$

1.5.6 ヤコビ記号

平方剰余記号を拡張し **ヤコビ記号** が次の様に定義される。正の奇数 q の素因数分解を $q = p_1^{e_1} \cdots p_r^{e_r}$ とするとき、q と整数 a に対して

$$\left(\frac{a}{q}\right) := \left(\frac{a}{p_1}\right)^{e_1} \cdots \left(\frac{a}{p_r}\right)^{e_r}$$

(q が素数のときは平方剰余記号に等しい。)

1.5.7 命題

ヤコビ記号には次の性質があり、これらを用いて高速に計算することができる。
(a, b は整数、q, r は正の奇数)

(1) $a \equiv b \ (\bmod q)$ ならば $\left(\frac{a}{q}\right) = \left(\frac{b}{q}\right).$

(2) a が q と互いに素ならば $\left(\frac{a^2}{q}\right) = 1.$ 特に $\left(\frac{1}{q}\right) = 1.$

(3) $\left(\frac{ab}{q}\right) = \left(\frac{a}{q}\right)\left(\frac{b}{q}\right).$

(4) $\left(\frac{-1}{q}\right) = \begin{cases} 1 & q \equiv 1 \ (\bmod 4) \text{ のとき} \\ -1 & q \equiv 3 \ (\bmod 4) \text{ のとき} \end{cases}$ (第 1 補充法則)

(5) $\left(\dfrac{2}{q}\right) = \begin{cases} 1 & q \equiv 1,7 \ (\bmod 8) \text{ のとき} \\ -1 & q \equiv 3,5 \ (\bmod 8) \text{ のとき} \end{cases}$ （第 2 補充法則）

(6) $\left(\dfrac{r}{q}\right) = \begin{cases} \left(\dfrac{q}{r}\right) & q \equiv 1 \ (\bmod 4) \text{ または } r \equiv 1 \ (\bmod 4) \text{ のとき} \\ -\left(\dfrac{q}{r}\right) & q \equiv r \equiv 3 \ (\bmod 4) \text{ のとき} \end{cases}$ （相互法則）

1.5.8 例

ヤコビ記号の中の数が小さくなってゆくように (1–6) の公式を工夫して使う。最後は (2) か第 1・2 補充法則が使える形になる。

$$\left(\frac{7}{19}\right) = -\left(\frac{19}{7}\right) = -\left(\frac{5}{7}\right) = -\left(\frac{7}{5}\right) = -\left(\frac{2}{5}\right) = -(-1) = 1$$

1.5.9 サンプルプログラム

Pascal で書いたヤコビ記号の関数定義部を以下に示す。

```
function jacobi_symbol(a,b:integer):integer;
var c,j:integer;
begin
  j:=1;
  if a<0 then begin a:=-a; if (b mod 4)=3 then j:=-j end;
  a:=a mod b;
  while a>1 do
    begin
      while (a mod 2)=0 do
        begin
          a:=a div 2;
          if ((b mod 8)=3) or ((b mod 8)=5) then j:=-j
        end;
      if a<>1 then
        begin
          if ((a mod 4)=3) and ((b mod 4)=3) then j:=-j;
          c:=b; b:=a; a:=c mod b
        end
      end;
  if a=0 then j:=0;
  jacobi_symbol:=j
end;
```

2　公開鍵暗号

2.1　古典的暗号と公開鍵暗号

2.1.1　状況設定

P = 平文 (ひらぶん、plain text) の集合

C = 暗号文 (cipher text) の集合

$E : P \longrightarrow C$　全単射

$D = E^{-1} : C \longrightarrow P$　E の逆写像

のとき、

$$P \xrightarrow{E} C \xrightarrow{D} P$$

のシステムを **暗号** と言い、

$$\begin{cases} E \text{ を暗号化関数 (encryption)} \\ D \text{ を復号化関数 (decryption)} \end{cases}$$

と言う。更に

$$K = \text{鍵の集合}$$

があって、各 $A \in K$ に対して暗号

$$P \xrightarrow{E_A} C \xrightarrow{D_A} P$$

が定まっているとき、この暗号の族

$$\left\{ P \xrightarrow{E_A} C \xrightarrow{D_A} P \right\}_{A \in K}$$

を **暗号系** と言う。

2.1.2　シーザー暗号系

$$P = C = \{ \sqcup, A, B, \cdots, Z \} = \text{空白と大文字のアルファベット合計 27 文字の集合}$$

を、この要素の順番で

$$\mathbf{Z}/27\mathbf{Z} = \{ 0, 1, 2, \cdots, 26 \}$$

と同一視する。鍵の集合を

$$K = \mathbf{Z}/27\mathbf{Z} - \{ 0 \}$$

とし、各 $n \in K$ に対して

$$E_n(x) = x + n \qquad (\text{ in } \mathbf{Z}/27\mathbf{Z})$$

と定められる暗号 $E_n : P \longrightarrow C$ を **シーザー暗号** と言う。アルファベットとしては E_n は n 文字先のアルファベットに置き換えることを意味する。例えば、

$$\texttt{INFORMATION} \quad \xrightarrow{E_3} \quad \texttt{LQIRUPDWLRQ}$$

$D_n(y) = y - n$ であるので、シーザー暗号は暗号化鍵 n がわかれば直ちに復号化関数がわかってしまう。

2.1.3　定義

この様に暗号化鍵を公開すると復号化関数がわかってしまう暗号系を **古典的暗号系** と言う。

2.1.4 定義

これに対し、次の様な暗号系を **公開鍵暗号系** と言う。

(1) 各鍵 $A \in K$ は **公開鍵**、**秘密鍵** と呼ばれるふたつの部分から成る：

$$A = (A_p, A_s) \qquad A_p: \text{ 公開鍵}, \quad A_s: \text{ 秘密鍵}$$

(2) 暗号化関数 E_A は公開鍵 A_p のみを用いて計算できる。

(3) 復号化関数 D_A は公開鍵 A_p がわかっただけでは事実上計算が不可能で、秘密鍵 A_s を用いて初めて計算できる。

2.1.5 古典的暗号の使い方

送信者と受信者が予め鍵 $A \in K$ を打ち合わせ、互いに秘密にする。

2.1.6 公開鍵暗号の使い方

(1) 受信者は鍵 $A = (A_p, A_s)$ を作成し、公開鍵 A_p を公開する。(秘密鍵 A_s は自分だけが持っている。)

(2) 送信者は公開鍵 A_p を用いて平文 x を暗号文 $y = E_A(x)$ に変換して受信者に送信する。

(3) 受信者は秘密鍵 A_s を用いて暗号文 y から $x = D_A(y)$ を計算する。

2.1.7 公開鍵暗号の利点

(1) 古典的暗号は鍵の打ち合わせをする必要があり、その際に鍵を盗まれる可能性がある。

(2) N 人の人間が暗号通信を行うとき、古典的暗号は $\frac{N(N-1)}{2}$ 個の鍵を必要とするのに対し、公開鍵暗号は N 個の鍵を用意すればよく、鍵を作るコストが少なくて済む。

2.2 デジタル署名

通信文に署名を付けるにはどうすれば良いか？

2.2.1 仮定

送信者 (受信者ではなく) の用いる公開鍵暗号

$$P \xrightarrow{E_A} C \xrightarrow{D_A} P$$

が

$$P = C, \qquad E_A \circ D_A = \mathrm{id}_C \text{ (恒等写像)}$$

を満たすとする。(のちに述べる RSA 暗号などはこの条件を満たしている。)

2.2.2 デジタル署名の手順

通信文を x, 送信者の鍵を $A = (A_p, A_s)$ とする。

Step 1 : 送信者は秘密鍵 A_s を用いて $y = D_A(x)$ を計算する。

Step 2 : 送信者は x と y を組にして (x, y) を送信する。

Step 3 : 受信者は送信者の公開鍵を用いて $E_A(y)$ を計算し、$x = E_A(y)$ が成り立つことを検証する。

2.2.3 キーポイント

秘密鍵 A_s を持たない者は $x = E_A(y)$ を満たす y を計算することができない。

2.2.4 注意

「署名」と言っても、名前の部分 $name$ だけを $D_A(name)$ と加工して $(x, D_A(name))$ を送信してはいけない。盗聴した第三者が $D_A(name)$ の部分だけを切り取って使うことができるからである。

3 RSA 暗号

3.1 RSA 暗号の原理

3.1.1 記号

$x \in \mathbf{Z}$ に対し $(x, \mathbf{mod}\ n)$ は法 n での x の剰余を

$$0 \leqq (x, \mathbf{mod}\ n) < n$$

の範囲に取ったものを表すこととする。

3.1.2 状況設定

p, q : 大きな異なる素数

$n = pq$

$m = \varphi(n) = (p-1)(q-1)$

e : m と互いに素な自然数

d : $ed \equiv 1 \pmod{m}$ を満たす自然数

とする。p, q, e を決めると n, m, d は自動的に定まる。(特に d はユークリッドのアルゴリズムによって高速に求まる。) 以上の記号のもと、

平文・暗号文の集合 : $P = C = \{\, x \in \mathbf{Z} \,|\, 0 \leqq x < n \,\}$

公開鍵 : $A_p = (n, e)$

秘密鍵 : $A_s = (p, q, m, d)$

とする。

3.1.3 暗号化

送信者は公開鍵 n, e を用いて通信文 x を

$$y = E_A(x) := (x^e, \mathbf{mod}\ n)$$

と暗号化して送信する。

3.1.4 復号化

受信者は秘密鍵 d を用いて受信文 y から

$$w = D_A(y) := (y^d, \mathbf{mod}\ n)$$

を計算すると $x = w$ となり通信文 x を得る。

3.1.5 証明

x が p で割り切れるとき

$$w \equiv x^{ed} \equiv 0 \equiv x \qquad (\bmod p)\,.$$

x が p で割り切れないときは $ed \equiv 1 \pmod{p-1}$ ゆえ、フェルマーの小定理よりやはり

$$w \equiv x^{ed} \equiv x \qquad (\bmod p)\,.$$

法 q に対しても同様で、中国剰余定理により

$$w \equiv x^{ed} \equiv x \qquad (\bmod n)\,.$$

w も x も $0 \leqq x, w < n$ の範囲にあって法 n での剰余が一致するので $w = x$. □

3.2 RSA 暗号が安全と考えられている理由

3.2.1 RSA 暗号が安全と考えられている理由

(1) n, e が公開されていても、y から

$$y = (x^e, \mathbf{mod}\ n)$$

を満たす x を計算すること (離散対数問題) は手に負えない。

(2) d がわかれば復号化関数がわかるが、次の命題により d がわかることと n の素因数 p, q を知ることは同値であり、それは手に負えないと信じられている。

3.2.2 命題

次は同値である。

(1) p, q がわかる。

(2) m がわかる。

(3) d がわかる。

3.2.3 補題

$$\begin{cases} a^2 \equiv b^2 \pmod{n} \\ a \not\equiv \pm b \pmod{n} \end{cases}$$

を満たす 2 整数 a, b がみつかれば p, q が求まる。

3.2.4 証明

$$(a+b)(a-b) \equiv 0 \pmod{n}, \qquad a \pm b \not\equiv 0 \pmod{n}$$

ゆえ、n のふたつの素因数 p, q は $a+b$ と $a-b$ の片方ずつに分れている。従ってユークリッドのアルゴリズムを用いて $(a+b, n)$ を計算すれば n の素因数が求まる。□

3.2.5 系

$$\begin{cases} a^2 \equiv 1 \pmod{n} \\ a \not\equiv \pm 1 \pmod{n} \end{cases}$$

を満たす 2 整数 a, b がみつかれば p, q が求まる。

3.2.6 命題 3.2.2 の証明

(3) $\Rightarrow$ (1) のみ概略を述べる (他は易しい)。

$$ed - 1 = 2^s t \qquad (\ t \text{ は奇数}\ )$$

と置く。n と互いに素な w をランダムに発生させると

$$w^{2^s t} \equiv 1 \pmod{n}$$

である。

$$w^t, w^{2t}, \cdots, w^{2^s t}$$

の中で最初に $\equiv 1 \pmod{n}$ となるを $w^{2^r t}$ とするとき

$$r = 0 \qquad \text{または} \qquad w^{2^{r-1} t} \equiv -1 \pmod{n}$$

となる確率は高々 50% であることがわかる (合同式の解を数える)。従って充分な個数の w を検査すれば

$$w^{2^{r-1} t} \not\equiv -1 \pmod{n}, \qquad w^{2^r t} \equiv 1 \pmod{n}$$

を満たす w がみつかり、系 **3.2.5** の a として $a = w^{2^{r-1} t}$ を取ることができる。□

3.3 危険な鍵

3.3.1 p q のときは危ない

$x = \dfrac{p+q}{2}$, $y = \left| \dfrac{p-q}{2} \right|$ とおくと

$$x \quad \sqrt{n}, \qquad y \quad 0, \qquad x^2 - y^2 = n$$

が成り立つ。従って $b = 1, 2, \cdots$ の順に $b^2 + n$ が平方数 ($= a^2$) となるまで検索すれば、b が y に達するまでの間に命題 **3.2.3** の条件を満たす a, b がみつかる。y は小さいのでその検索時間は小さい。

3.3.2 $p-1$ と $q-1$ が大きな公約数を持つときは危ない

$p-1$ と $q-1$ の最小公倍数を ℓ とするとき、

$$ed' \equiv 1 \pmod{\ell}$$

を満たす d' も復号化指数として用いることができる (**3.1.5** 参照)。$p-1$ と $q-1$ が大きな公約数を持てば ℓ は比較的小さくなり、d' を検索によってみつけられる可能性が大きくなる。

3.3.3 $\varphi(n)$ が小さな素因数しか持たないときは危ない

$K =$(小さな素数のある程度のべきの積) と置くと、$\varphi(n)$ が小さな素因数しか持たないときは K は $\varphi(n)$ の倍数になる。そこで

$$ed' \equiv 1 \pmod{K}$$

を満たす d' を求めれば、d' も復号化指数として用いることができる。

3.4 鍵の大きさ

3.4.1 現行の鍵の大きさ

アメリカ合衆国が規制を掛けていて、n は 512 ビット (10 進数で約 155 桁) 以下でなければならない。他方、

3.4.2 素因数分解の現状

'93) 10 進 120 桁の RSA チャレンジ数 RSA-120 が 2 次ふるい法を用いて 825MIPSYear で素因数分解された。

'94) 10 進 129 桁の RSA チャレンジ数 RSA-129 が 2 次ふるい法を用いて 5000MIPSYear で素因数分解された。

'95) 10 進 130 桁の RSA チャレンジ数 RSA-130 を数体ふるい法を用いて素因数分解するプロジェクトがスタートした。

4 平方根の話

4.1 開平法

4.1.1 例

$\sqrt{34567}$ を求めよう。

(1) 34567 を、小数点から 2 桁ずつ区切る。

```
  ___________
√ 3 45 67
```

(2) a^2 が最初の 3 を越えないような a（ $=1$ ）を定め、

(a) $\sqrt{\ }$ の上に a を書き、

(b) 3 の下に a^2（ $=1$ ）を、差（ $=2$ ）を線の下に書いて、次の 2 桁（ $=45$ ）を下ろし、

(c) 左に a（ $=1$ ）を 2 段重ねて書いて和（ $=2$ ）を線の下に書く。

```
1          1
1       √ 3 45 67
1          1
--      ----------
2          2 45
```

(3) 左にある 2 に注目して、

(a) $(20+b)\times b$ が 245 を越えないような b（ $=8$ ）を定め、

(b) $\sqrt{\ }$ の上に b を書き、

(c) 245 の下に $(20+b)\times b$（ $=224$ ）を、差（ $=21$ ）を線の下に書き、次の 2 桁（ $=67$ ）を下ろし、

(d) 左に $20+b$（ $=28$ ）と b（ $=8$ ）を 2 段重ねて書いて和（ $=36$ ）を線の下に書く。

```
1          1  8
1       √ 3 45 67
1          1
--      ----------
28         2 45
 8         2 24
--      ----------
36           21 67
```

(4) 以下同様。

```
1          1  8  5.
1       √ 3 45 67
1          1
--      ----------
28         2 45
 8         2 24
--      ----------
365          21 67
  5          18 25
--      ----------
370           3 42  00
```

⋮

		1	8	5.	9	2	2	0	2
1	√	3	45	67					
1		1							
28		2	45						
8		2	24						
365			21	67					
5			18	25					
3709			3	42	00				
9			3	33	81				
37182				8	19	00			
2				7	43	64			
371842					75	36	00		
2					74	36	84		
3718440						99	16	00	
0								0	
37184402						99	16	00	00
2						74	36	88	04
37184404						24	79	11	96

4.1.2 仕組み

1 桁目

$$(100a)^2 \leqq 34567$$

$$a^2 \leqq 3$$

$$a = 1$$

2 桁目

$$(100 + 10b)^2 \leqq 34567$$

$$(10 + b)^2 \leqq 345$$

$$10^2 + 20b + b^2 \leqq 345$$

$$(20 + b) \times b \leqq 345 - 10^2 = 245$$

$$b = 8$$

3 桁目

$$(180 + c)^2 \leqq 34567$$

$$180^2 + 360c + c^2 \leqq 34567$$

$$(360 + c) \times c \leqq 34567 - 100^2 - (200 + 80) \times 80 = 2167$$

$$c = 5$$

4 **桁目**

$$(185 + d/10)^2 \leqq 34567$$

$$1850^2 + 3700d + d^2 \leqq 3456700$$

$$(3700 + d) \times d \leqq 3456700 - 1000^2 - (2000 + 800) \times 800 - (3600 + 50) \times 50 = 34200$$

$$d = 9$$

以下同様

4.1.3 計算量

1 つの桁を求めるのに、高々3 回程度の試行 (積の計算とと大小比較) と 1 回の引き算を行えば良い。

4.2 平方根を求めるニュートン法

$\sqrt{\mathrm{a}}$ を求めよう。

4.2.1 アルゴリズム

次のように数列 $\{ x_n \}_{n=0,1,2,\cdots}$ を計算すると

$$x_n \longrightarrow \sqrt{a} \qquad (n \to \infty)$$

となる。

(1) $x_0 :=$ (適当な初期値) と置く。

(2) $x_{n+1} := \dfrac{1}{2}\left(x_n + \dfrac{a}{x_n}\right) \qquad (n = 0, 1, 2, \cdots)$

4.2.2 注意

このアルゴリズムは多倍長の整数 a に対しても有効で、高速に $\sqrt{a}$ を計算する。

4.2.3 注意

開平法もニュートン法も、実数の大小概念がキーポイントになっている。

4.3 法演算での平方根

4.3.1 定理

奇素数 p を法とする平方剰余は丁度 $\dfrac{p-1}{2}$ 個あり、それらは $(\mathbf{Z}/p\mathbf{Z})^\times$ の指数 2 の部分群を成す。

4.3.2 証明

g を $\bmod p$ の原始根とすると、

$$\{ x \in (\mathbf{Z}/p\mathbf{Z})^\times \mid x \text{ は平方剰余} \} = \{ g^0, g^2, g^4, \cdots \}$$

ゆえ。群論の言葉で言えば、平方剰余記号

$$(\mathbf{Z}/p\mathbf{Z})^\times \longrightarrow \{ \pm 1 \}\,; \qquad a \mapsto \left(\frac{a}{p}\right)$$

は乗法群の全射準同型であり、その核が $\{ x \in (\mathbf{Z}/p\mathbf{Z})^\times \mid x$ は平方剰余 $\}$ である。

4.3.3 注意

素数 p を法とするとき、

$$a^2 \equiv b^2 \ (\bmod p) \implies a \equiv b \ (\bmod p) \ \text{または} \ a \equiv -b \ (\bmod p) .$$

特に、

$$a^2 \equiv 1 \ (\bmod p) \implies a \equiv 1 \ (\bmod p) \ \text{または} \ a \equiv -1 \ (\bmod p) .$$

4.3.4 法演算での平方根を求めるアルゴリズム（ 法が奇素数の場合 ）

奇素数 p を法とし、$x^2 \equiv a \ (\bmod p)$ をみたす x を求める。

(1) 平方剰余記号を計算して、a が平方剰余でなければ終了。

(2) $p = 2u + 1$（ u は奇数 ）のとき $x := \pm a^{\frac{u+1}{2}}$.

(3) $p = 4u + 1$（ u は奇数 ）のとき

(3a) $a^u \equiv 1 \ (\bmod p)$ のとき $x := \pm a^{\frac{u+1}{2}}$.

(3b) $a^u \equiv -1 \ (\bmod p)$ のとき $x := \pm a^{\frac{u+1}{2}} \times 2^u$.

(4) $p = 8u + 1$（ u は奇数 ）のとき

(4a) 奇素数 $b = 3, 5, 7, 11, \cdots$ を順番に検索して $\left(\dfrac{b}{p}\right) = -1$ をみたす b をひとつ求める。

(4b) $a^{2u} \equiv 1 \ (\bmod p)$ のとき

(4b-1) $a^u \equiv 1 \ (\bmod p)$ のとき $x := \pm a^{\frac{u+1}{2}}$.

(4b-2) $a^u \equiv -1 \ (\bmod p)$ のとき $x := \pm a^{\frac{u+1}{2}} \times b^{2u}$.

(4c) $a^{2u} \equiv -1 \ (\bmod p)$ のとき

(4c-1) $a^u \times b^{2u} \equiv 1 \ (\bmod p)$ のとき $x := \pm a^{\frac{u+1}{2}} \times b^u$.

(4c-2) $a^u \times b^{2u} \equiv -1 \ (\bmod p)$ のとき $x := \pm a^{\frac{u+1}{2}} \times b^{3u}$.

(5) 一般に $p = 2^k u + 1$（ $k \geqq 3$, u は奇数 ）のとき

(5a) 奇素数 $b = 3, 5, 7, 11, \cdots$ を順番に検索して $\left(\dfrac{b}{p}\right) = -1$ をみたす b をひとつ求める。

(5b) $i = 1, 2, \cdots, k-1$ に対して $e_i = 0$ または 1 を次のように定める：

(5b-1) $a^{2^{k-i-1}u} \times b^{(2^{k-i}e_1 + \cdots + 2^{k-2}e_{i-1})u} \equiv 1 \ (\bmod p)$ のとき $e_i := 0$,

(5b-2) $a^{2^{k-i-1}u} \times b^{(2^{k-i}e_1 + \cdots + 2^{k-2}e_{i-1})u} \equiv -1 \ (\bmod p)$ のとき $e_i := 1$.

(5c) $x := \pm a^{\frac{u+1}{2}} \times b^{(e_1 + 2e_2 + \cdots + 2^{k-2}e_{k-1})u}$ と置く。

4.3.5 例

(2) $p = 23 = 2 \times 11 + 1$（ $u = 11$ ）, $a = 2$ の場合、$x = \pm 2^{\frac{u+1}{2}} = \pm 2^6 \equiv \mp 5$.（ 実際、$(\mp 5)^2 = 25 \equiv 2$. ）

(3a) $p = 29 = 4 \times 7 + 1$（ $u = 7$ ）, $a = 7$ の場合、$7^u = 7^7 \equiv 1$ より、$x = \pm 7^{\frac{u+1}{2}} = \pm 7^4 \equiv \mp 6$.（ 実際、$(\mp 6)^2 = 36 \equiv 7$. ）

(3b) $p = 29 = 4 \times 7 + 1$（ $u = 7$ ）, $a = 5$ の場合、$5^u = 5^7 \equiv -1$ より、$x = \pm 5^{\frac{u+1}{2}} \times 2^7 \equiv \mp 11$.（ 実際、$(\mp 11)^2 = 121 \equiv 5$. ）

4.3.6 アルゴリズムの証明

ここでは法は p とする。

(2) **の場合** オイラーの規準 **1.5.4** より

$$a^u = a^{\frac{p-1}{2}} \equiv \left(\frac{a}{p}\right) = 1.$$

$$\therefore \quad a \equiv a^{u+1} = \left(a^{\frac{u+1}{2}}\right)^2.$$

(3) **の場合** オイラーの規準より

$$a^{2u} = a^{\frac{p-1}{2}} \equiv \left(\frac{a}{p}\right) = 1.$$

4.3.3 より

$$a^u \equiv 1 \quad \text{または} \quad a^u \equiv -1$$

(3a) $a^u \equiv 1$ ならば

$$a \equiv a^{u+1} = \left(a^{\frac{u+1}{2}}\right)^2.$$

(3b) $a^u \equiv -1$ ならば、オイラーの規準より

$$2^{2u} = 2^{\frac{p-1}{2}} \equiv \left(\frac{2}{p}\right) = -1.$$

$$\therefore \quad a^u \times 2^{2u} \equiv 1$$

$$\therefore \quad a \equiv a^{u+1} \times 2^{2u} = \left(a^{\frac{u+1}{2}} \times 2^u\right)^2.$$

(4) **の場合** オイラーの規準より

$$a^{4u} = a^{\frac{p-1}{2}} \equiv \left(\frac{a}{p}\right) = 1.$$

4.3.3 より

$$a^{2u} \equiv 1 \quad \text{または} \quad a^{2u} \equiv -1$$

(4a) $a^{2u} \equiv 1$ ならば、**4.3.3** より

$$a^u \equiv 1 \quad \text{または} \quad a^u \equiv -1$$

(4a-1) $a^u \equiv 1$ ならば

$$a \equiv a^{u+1} = \left(a^{\frac{u+1}{2}}\right)^2.$$

(4a-2) $a^u \equiv -1$ ならば、オイラーの規準より

$$b^{4u} = b^{\frac{p-1}{2}} \equiv \left(\frac{2}{p}\right) = -1.$$

$$\therefore \quad a^u \times b^{4u} \equiv 1$$

$$\therefore \quad a \equiv a^{u+1} \times b^{4u} = \left(a^{\frac{u+1}{2}} \times b^{2u}\right)^2.$$

(4b) $a^{2u} \equiv -1$ ならば、

$$a^{2u} \times b^{4u} \equiv 1.$$

4.3.3 より

$$a^u \times b^{2u} \equiv 1 \quad \text{または} \quad a^u \times b^{2u} \equiv -1$$

(4b-1) $a^u \times b^{2u} \equiv 1$ ならば

$$a \equiv a^{u+1} \times b^{2u} = (a^{\frac{u+1}{2}} \times b^u)^2.$$

(4b-2) $a^u \times b^{2u} \equiv -1$ ならば、

$$a^u \times b^{2u} \times b^{4u} \equiv 1.$$

$$\therefore \quad a \equiv a^{u+1} \times b^{6u} = (a^{\frac{u+1}{2}} \times b^{3u})^2.$$

(5) **の場合** は (4) の一般化。

4.3.7 法演算での平方根を求めるアルゴリズム (法が奇素数のべきの場合)

奇素数のべき p^e を法とし、$x^2 \equiv a \pmod{p^e}$ をみたす x を求める。

(1) 平方剰余記号を計算して、$\left(\frac{a}{p}\right) = 1$ でなければ終了。

(2) $(x_1)^2 \equiv a \pmod{p}$ の解 x_1 をアルゴリズム **4.3.4** によって求める。

(3) $(x_e)^2 \equiv a \pmod{p^e}$ の解 x_e ($e = 2, 3, \cdots$) を次の式で順次求める。

$$x_e := x_{e-1} + (2x_{e-1})^{-1} \times (a - (x_{e-1})^2).$$

ただし $(2x_{e-1})^{-1}$ は mod p での $2x_{e-1}$ の逆数を表す (ユークリッドのアルゴリズムを用いて計算する) 。

4.3.8 証明

$a - (x_{e-1})^2 \equiv 0 \pmod{p^{e-1}}$ ゆえ、

$$(a - (x_{e-1})^2)^2 \equiv 0 \pmod{p^{2e-2}}.$$

$2e - 2 = e + (e-2) \geqq e$ ゆえ

$$(a - (x_{e-1})^2)^2 \equiv 0 \pmod{p^e}.$$

従って

$$\begin{aligned}
&(x_{e-1} + (2x_{e-1})^{-1} \times (a - (x_{e-1})^2))^2 \\
&\quad = \quad (x_{e-1})^2 + (a - (x_{e-1})^2) + ((2x_{e-1})^{-1})^2 \times (a - (x_{e-1})^2)^2 \\
&\quad \equiv \quad (x_{e-1})^2 + (a - (x_{e-1})^2) \\
&\quad = \quad a \qquad \pmod{p^e}
\end{aligned}$$

4.3.9 法演算での平方根を求めるアルゴリズム (法が奇数の合成数の場合)

奇数 n を法とし、$x^2 \equiv a \pmod{n}$ をみたす x を求める。

(1) $n = p_1^{e_1} \cdots p_m^{e_m}$ と素因数分解する。

(2) 各素因子 p_j について平方剰余記号を計算して、ひとつでも $\left(\frac{a}{p_j}\right) = 1$ でなければ終了。

(3) $(x_j)^2 \equiv a \pmod{p_j^{e_j}}$ の解 x_j をアルゴリズム **4.3.7** によって求める。

(4) 中国剰余定理を用いて、連立合同式

$$x \equiv x_j \qquad \pmod{p_j^{e_j}} \qquad (j = 1, 2, \cdots, m)$$

を解く。

4.3.10 定理

n が奇数の法のとき、次は同値。

(1) n の素因数分解がわかる。

(2) 2 次合同式 $x^2 \equiv a \pmod{n}$ が解を持つとき、その解をすべて計算できる。

4.3.11 証明

(1) ⇒ (2) は上に述べたとおり。(2) ⇒ (1) を示す。n の素因数 p をひとつ取り $n = p^e m$ ($(p, m) = 1$) と置くとき、$x^2 \equiv a \pmod{n}$ の解として

$$x_1 \equiv -x_2 \pmod{p^e}, \qquad x_1 \equiv x_2 \pmod{m}$$

なる 2 数 x_1, x_2 が存在する。このとき補題 **3.2.3** と同じ議論によって、ユークリッドのアルゴリズムを用いて $p^e = (x_1 + x_2, n)$ が求まる。□

5 ゼロ知識証明

5.1 Fiat-Shamir 認証

5.1.1 状況設定

登場人物：

クレジットカード会社 A, 証明者 (クレジットカード利用者) P, 検証者 (販売店) V

登場する数：

p, q : カード会社 A だけが知っている素数 ,

$n = pq$: 公開 ,

t : 証明者 P の パスワード (非公開) ,

$a = (t^2, \ \mathbf{mod}\ n)$: 証明者 P の ID 番号 (公開)

目 的：

証明者 P がパスワード t を知っていることを、t についての一切の情報を漏らすことなく、検証者 V に証明する。

5.1.2 アイデア

定理 **4.3.10** によって、クレジットカード会社 A と証明者 P 以外は t を計算できないことを利用する。

5.1.3 プロトコル A

Step 1 : 証明者 P は乱数 r を選び、$x := (r^2, \mathbf{mod}\ n)$ を検証者 V に送る。

Step 2 : 検証者 V は $b = 0$ または 1 をランダムに選び、b を証明者 P に送る。

Step 3 : b に応じて、証明者 P は

$$y := \begin{cases} r & b = 0 \text{ の場合} \\ tr & b = 1 \text{ の場合} \end{cases}$$

を検証者 V に送る。

Step 4 : 検証者 V は次式が成立するか否かを検証する :

$$\begin{cases} x \equiv y^2 \ (\mathrm{mod}\ n) & b = 0 \text{ の場合} \\ ax \equiv y^2 \ (\mathrm{mod}\ n) & b = 1 \text{ の場合} \end{cases}$$

以上の操作を充分な回数繰り返す。

5.1.4 プロトコル A の仕組み

(1) 検証者 V から常に $b = 0$ が送られて来るならば、偽証明者 P′ は t を知らなくて良い。

(2) 検証者 V から常に $b = 1$ が送られて来るとわかっていれば、偽証明者 P′ は先に y をランダムに選び、**Step 1** で $x := y^2/a$ を検証者 V に送ることによって (t を知らないにもかかわらず)

$$ax \equiv y^2 \ (\mathrm{mod}\ n)$$

を成り立たせることができる。

(3) すなわち、偽証明者 P′ が検証者 V を騙すためには

$$\begin{cases} b = 0 \text{ を予測} & \Rightarrow \quad \text{乱数 } r \text{ を先に選ぶ} \\ b = 1 \text{ を予測} & \Rightarrow \quad \text{乱数 } y \text{ を先に選ぶ} \end{cases}$$

という作戦を取らなければならない。

(4) しかし b が予測できない状況で試行を m 回繰り返せば、偽証明者 P′ が検証者 V を騙し得る確率は $\left(\frac{1}{2}\right)^m \to 0$ ($m \to \infty$) である。

5.2 RSA 暗号の秘密鍵保持者であることのゼロ知識証明

5.2.1 状況設定

登場人物：

証明者 (RSA 暗号の秘密鍵保持者) P, 検証者 V

登場する数：

p, q : 証明者 P だけが知っている RSA 暗号の秘密鍵 ,

$n = pq$: 公開鍵

目 的：

証明者 P が秘密鍵 p, q を知っていることを、秘密鍵についての一切の情報を漏らすことなく、検証者 V に証明する。

5.2.2 アイデア

定理 **4.3.10** によれば、証明者 P は任意に与えられた $\bmod n$ の平方剰余 a に対して $t^2 \equiv a \pmod{n}$ を満たす t を計算できることを示せば良い。ただし t についての情報を漏らすことは、秘密鍵についての情報を漏らしてしまうので避けなければならない。

5.2.3 プロトコル案 1

Step 1 : 検証者 V は s をランダムに選び、$a := (s^2, \mathbf{mod}\ n)$ を証明者 P に送る。

Step 2 : 証明者 P は $t^2 \equiv a \pmod{n}$ を満たす t を計算し、t を検証者 V に送る。

Step 3 : 検証者 V は

$$a \equiv t^2 \pmod{n}$$

が成立するか否かを検証する。

5.2.4 プロトコル案 1 の欠点

検証者 V の持っている s と t の間に

$$s \equiv t \pmod{p}, \qquad s \equiv -t \pmod{q}$$

という関係が確率 1/2 で成立する。このとき

$$p = (s - t, n)$$

となるので、ユークリッドのアルゴリズムによって p が検証者 V にわかってしまう。

5.2.5 プロトコル案 2

Step 1 : 検証者 V は s をランダムに選び、$a := (s^2, \mathbf{mod}\ n)$ を証明者 P に送る。

Step 2 : 証明者 P は $t^2 \equiv a \pmod{n}$ を満たす t を計算する。

Step 3 : 証明者 P はプロトコル A を用いて t を持っていることを検証者 V に証明する。

5.2.6 プロトコル案 2 の欠点

不正な検証者 V′ は、平方剰余かどうかわからない乱数 a を証明者 P に送ることによって、証明者 P を「平方剰余性判定マシン」として利用することができる。(平方剰余性を判定することで秘密情報が漏れることになる。)

5.2.7 プロトコル B

Step 1 : 検証者 V は s をランダムに選び、$a := (s^2, \mathbf{mod}\ n)$ を証明者 P に送る。

Step 2 : 検証者 V はプロトコル A を用いて s を持っていることを証明者 P に証明する。

Step 3 : 証明者 P は $t^2 \equiv a \pmod{n}$ を満たす t を計算する。

Step 4 : 証明者 P はプロトコル A を用いて t を持っていることを検証者 V に証明する。

5.3 通信によるコイントス

5.3.1 状況設定

ふたりの人間 A と B が通信によってコイントスを行う。

5.3.2 定義

p, q を $p \equiv q \equiv 3 \pmod{4}$ を満たす素数として、$n = pq$ と表される数を Blum 数という。

5.3.3 命題

$n = pq$ を Blum 数、a は n と互いに素な mod n の平方剰余とする。このとき

(1) $x^2 \equiv a \bmod n$ の解 x は mod n で 4 個存在して $\pm x_1, \pm x_2$ と書け、それぞれヤコビ記号が

$$\left(\frac{\pm x_1}{n}\right) = 1, \qquad \left(\frac{\pm x_2}{n}\right) = -1$$

を満たす。

(2) (1) の両方の解を求めることは、n の素因数分解を計算することと同程度に難しい。

5.3.4 プロトコル C

Step 1 : A は $p \equiv q \equiv 3 \pmod{4}$ を満たす大きな素数 p, q を選び、$n = pq$ を B に送る。

Step 2 : B は x をランダムに選び、$a := (x^2, \mathbf{mod}\ n)$ を A に送る。

Step 3 : A は $e = -1$ または 1 をランダムに選び、e を B に送る。

Step 4 : B は x を A に送る。

Step 5 : A は $x^2 \equiv a \pmod{n}$ を検証する。

Step 6 : $\left(\frac{x}{n}\right) = e$ か否かによって、A または B の勝ちとする。

5.3.5 プロトコル C の仕組み

(1) A は $x^2 \equiv a \pmod{n}$ の解をすべて求めることができるが、命題 **5.3.3** (1) により、$\left(\frac{x}{n}\right)$ を推定することはできない。

(2) 命題 **5.3.3** (2) により B は $x^2 \equiv a \pmod{n}$ を解くことができず、また

$$\left(\frac{-x}{n}\right) = \left(\frac{x}{n}\right)$$

なので、A から e を送られたあとで $\left(\frac{x'}{n}\right) = -e$ を満たす x' に取り替えることができない。

5.3.6 注意

(1) プロトコル C は、n が mod 4 で 3 余る素因数を偶数個持てば成立する。しかし、コストを考えると Blum 数が最良である。

(2) 逆に n が mod 4 で 1 余る素数 2 個の積ならば $x^2 \equiv a \pmod{n}$ の全ての解について $\left(\frac{x}{n}\right)$ は同符号になり、このとき A は B から送られた a からひとつの解 x を計算し、$e = \left(\frac{x}{n}\right)$ を B に送ることで確実に勝てる。

5.3.7 プロトコル C の続き

上の注意により、A は n が Blum 数であることを B に証明する必要がある。p や q を B に教えてしまうことはコストの無駄なので、次のプロトコルを用いる。

5.3.8 プロトコル D

Step 1 : A は x をランダムに選び、$a := (x^2, \mathbf{mod}\ n)$ を B に送る。

Step 2 : B は $e = -1$ または 1 をランダムに選び、e を A に送る。

Step 3 : A は $(x')^2 \equiv a \pmod{n}$ かつ $\left(\frac{x'}{n}\right) = e$ を満たす x' を計算し、x' を B に送る。

Step 4 : B は $(x')^2 \equiv a \pmod{n}$ と $\left(\frac{x'}{n}\right) = e$ を検証する。

以上の操作を充分な回数繰り返す。

5.4 グラフの同型を証明するゼロ知識証明

5.4.1 状況設定

登場人物 :

証明者 P と検証者 V

登場するグラフ :

G, H : 同型なふたつの有限無向グラフ (公開されている) ,

$\sigma : G \to H$: 同型写像 (証明者 P のみが知っている)

目 的 :

証明者 P が同型写像 σ を知っていることを、σ についての一切の情報を漏らすことなく、検証者 V に証明する。

5.4.2 同型写像の記述方法

G の頂点数 ($= H$ の頂点数) を n、G, H の隣接行列をそれぞれ $A = (a_{ij}), B = (b_{ij})$ とする。同型写像 σ は n 文字の置換であって、

$$a_{ij} = b_{\sigma(i)\sigma(j)} \qquad \text{for} \quad \forall i, j$$

を満たすものとして記述される。

5.4.3 プロトコル E

Step 1 : 証明者 P はランダムに n 文字の置換 ϕ を選び、$F = \phi(H)$ の隣接行列 $C = (c_{ij}) = (b_{\phi^{-1}(i)\phi^{-1}(j)})$ を検証者 V に送る。

Step 2 : 検証者 V は $e = 1$ または 2 をランダムに選び、e を証明者 P に送る。

Step 3 : e に応じて、証明者 P は置換

$$\tau := \begin{cases} \phi & e = 1 \text{ の場合} \\ \phi \circ \sigma & e = 2 \text{ の場合} \end{cases}$$

を検証者 V に送る。

Step 4 : 検証者 V は次が成立するか否かを検証する :

$$\begin{cases} F = \tau(H) \quad \text{すなわち} \quad c_{\tau(i)\tau(j)} = b_{ij} \quad \text{for} \quad \forall i, j & e = 1 \text{ の場合} \\ F = \tau(G) \quad \text{すなわち} \quad c_{\tau(i)\tau(j)} = a_{ij} \quad \text{for} \quad \forall i, j & e = 2 \text{ の場合} \end{cases}$$

以上の操作を充分な回数繰り返す。

5.4.4 プロトコル E の仕組み

(1) 検証者 V から常に $e = 1$ が送られて来るならば、偽証明者 P′ は σ を知らなくて良い。

(2) 検証者 V から常に $e = 2$ が送られて来るとわかっていれば、偽証明者 P′ は先に τ をランダムに選び、**Step 1** で $\tau(G)$ の隣接行列を検証者 V に送ることによって (σ を知らないにもかかわらず)

$$F = \tau(G)$$

を成り立たせることができる。

(3) すなわち、偽証明者 P′ が検証者 V を騙すためには

$$\begin{cases} e = 1 \text{ を予測} \quad \Rightarrow \quad \text{ランダムな置換 } \phi \text{ を先に選ぶ} \\ e = 2 \text{ を予測} \quad \Rightarrow \quad \text{ランダムな置換 } \tau \text{ を先に選ぶ} \end{cases}$$

という作戦を取らなければならない。

(4) しかし e が予測できない状況で試行を m 回繰り返せば、偽証明者 P′ が検証者 V を騙し得る確率は $\left(\frac{1}{2}\right)^m \to 0$ ($m \to \infty$) である。

6 素数判定法

6.1 フェルマーの小定理を利用した素数判定法

RSA 暗号や通信によるコイントスを設計する為には大きな素数を作る必要があった。以下 n をランダムに与えられた大きな奇数とし、n が素数かどうかを判定する方法を考える。

6.1.1 定理

n が素数ならば次の $(*)$ が成り立つ。

$$(*) \qquad a^{n-1} \equiv 1 \pmod{n} \qquad \text{for} \quad \forall a \in \mathbf{Z} \quad \text{s.t.} \quad (a,n) = 1$$

6.1.2 注

従って $a^{n-1} \not\equiv 1 \pmod{n}$ となる a がみつかれば n は素数でないことがわかる。しかし、$(*)$ が成り立っても n は素数とは言えない。

6.1.3 例

$n = 561 = 3 \times 11 \times 17$ とする。$(a,n) = 1$ ならば定理 **6.1.1** より

$$\begin{cases} a^2 \equiv 1 \pmod{3} \\ a^{10} \equiv 1 \pmod{11} \\ a^{16} \equiv 1 \pmod{17} \end{cases}$$

560 は 2, 10, 16 の公倍数ゆえ

$$a^{560} \equiv 1 \pmod{3 \times 11 \times 17}$$

6.1.4 定義

$(*)$ が成り立つような合成数 n をカーマイケル (Carmichael) 数と呼ぶ。

6.1.5 定理

奇数 n について次は同値。

(1) n はカーマイケル数。

(2) $n = p_1 p_2 \cdots p_r$ を n の素因数分解とするとき、

  (a) p_j は全て異なる

  (b) $r \geqq 3$

  (c) 各 j について $p_j - 1$ は $n - 1$ の約数。

6.1.6 定理

カーマイケル数は無限個存在する。(Alford, Graville and Pomerance (1992))

しかし $n-1$ が素因数分解できているときには $(*)$ を利用して素数判定ができる。

6.1.7 定理

$n-1=p_1^{e_1}p_2^{e_2}\cdots p_r^{e_r}$ と素因数分解できているとする。このとき、各 j に対して

$$\begin{cases} a_j^{n-1}\equiv 1 & (\bmod n) \\ a_j^{(n-1)/p_j}\not\equiv 1 & (\bmod n) \end{cases}$$

を満たす a_j がみつかれば、n は素数である。

6.1.8 証明

乗法群 $(\mathbf{Z}/n\mathbf{Z})^\times$ における a_j の位数を m_j とおくと、条件より m_j は $p_j^{e_j}$ で割り切れることがわかる。$(\mathbf{Z}/n\mathbf{Z})^\times$ の位数 $\varphi(n)$ は m_j の倍数ゆえ、

$$\varphi(n)\geqq (\ m_1,m_2,\cdots,m_r \text{ の最小公倍数 }) = p_1^{e_1}p_2^{e_2}\cdots p_r^{e_r}=n-1$$

したがって $\varphi(n)=n-1$ で n は素数。□

$n-1$ が完全に素因数分解できていなくても、次のような状況では同様に素数判定ができる。

6.1.9 定理

$n-1=p_1^{e_1}p_2^{e_2}\cdots p_r^{e_r}m$ (p_j は相異なる素数、$m<\sqrt{n}$) とする。このとき、各 j に対して

$$\begin{cases} a_j^{n-1}\equiv 1 & (\bmod n) \\ (a_j^{(n-1)/p_j}-1,n)=1 \end{cases}$$

を満たす a_j がみつかれば、n は素数である。

6.1.10 証明

n が合成数ならば $q\leqq\sqrt{n}$ なる素因数 q を持つ。条件より

$$\begin{cases} a_j^{n-1}\equiv 1 & (\bmod q) \\ a_j^{(n-1)/p_j}\not\equiv 1 & (\bmod q) \end{cases}$$

上の証明と同じ議論によって、a_j の $(\mathbf{Z}/q\mathbf{Z})^\times$ に於ける位数は $p_j^{e_j}$ の倍数となり、$\varphi(q)=q-1$ は $p_1^{e_1}p_2^{e_2}\cdots p_r^{e_r}=(n-1)/m$ の倍数となる。ところが

$$\frac{n-1}{m}>\frac{n-1}{\sqrt{n}}=\sqrt{n}-\frac{1}{\sqrt{n}}>\sqrt{n}-1\geqq q-1$$

矛盾が生じた。□

6.2 Solovay-Strassen の素数判定法

この方法のキーポイントはオイラーの規準である。

6.2.1 定理

n が素数ならば、任意の $a\in\mathbf{Z}$ に対して次の (※) が成り立つ。

(※) $$a^{\frac{n-1}{2}}\equiv\left(\frac{a}{n}\right) \qquad (\bmod n)$$

6.2.2 定理

n が素数でなければ、50% 以上の $a\in(\mathbf{Z}/n\mathbf{Z})^\times$ が (※) を満たさない。

6.2.3 証明

(1) まず、(※) を満たさない a がひとつ存在することを示す。

(a) n が平方因子を持たないとき。
n の素因子 p を取って $n = pm$ とし、mod p の平方非剰余 $a \in \mathbf{Z}$ を取る。中国剰余定理によって $a \equiv 1$ (mod m) として良い。すると

$$\left(\frac{a}{n}\right) = \left(\frac{a}{p}\right)\left(\frac{a}{m}\right) = -1, \qquad a^{\frac{n-1}{2}} \equiv 1 \quad (\text{ mod } m\ ).$$

(b) n が平方因子を持つとき。
素数 p の 2 乗が n を割り切るとし、$n = p^e m$ ($e \geqq 2, (p,m) = 1$) とおく。$a = 1 + \frac{n}{p} = 1 + p^{e-1}m$ とおけば、n の任意の素因子 q (p を含め) について $a \equiv 1$ (mod q) ゆえ

$$\left(\frac{a}{n}\right) = 1$$

他方、

$$a^{\frac{n-1}{2}} \equiv 1 + \frac{n-1}{2}p^{e-1}m \not\equiv 1 \quad (\text{ mod } p^e\ ).$$

(2) さて、(1) の a を固定する。$x \in (\mathbf{Z}/n\mathbf{Z})^\times$ が (※) を満たせば、ヤコビ記号の乗法性より ax は (※) を満たさないことがわかる。従って (※) を満たす $(\mathbf{Z}/n\mathbf{Z})^\times$ の元の個数は (※) を満たさない元の個数以下である。□

6.2.4 アルゴリズム

Step 1 : a ($1 < a < n$) をランダムに生成する。

Step 2 : $(a,n) > 1$ ならば (a,n) は n の真の約数となるので終了。

Step 3 : (※) をチェックする。

Step 4 : (※) が成り立たなければ終了、成り立てば **Step 1** へ戻る。

(※) が成り立つ限り、以上を充分な回数繰り返す。すると、合成数 n が (※) のチェックを m 回通過する確率は高々 $\left(\frac{1}{2}\right)^m$ となる。

6.3 Miller-Rabin の素数判定法

本節では $n-1$ を 2 で割れるだけ割って $n - 1 = 2^s t$ と置く。

6.3.1 定理

n が素数ならば任意の a ($(a,n) = 1$) に対して次が成り立つ :

$$(\sharp) \qquad \begin{cases} \text{(i)} \quad a^{\,t} \equiv 1 \qquad (\text{ mod } n\ ) \qquad \text{又は} \\ \text{(ii)} \quad \exists r\ (\ 0 \leqq r < s\ ) \quad \text{s.t.} \quad a^{2^r t} \equiv -1 \qquad (\text{ mod } n\ ) \end{cases}$$

6.3.2 証明

n が素数ならば $a^{2^s t} = a^{n-1} \equiv 1$ (mod n) で、かつ 1 の平方根は mod n で ± 1 のみであるから、

$$a^{2^{s-1}t} \equiv \pm 1 \qquad (\text{ mod } n\ )$$

$a^{2^{s-1}t} \equiv 1$ (mod n) ならば更に

$$a^{2^{s-2}t} \equiv \pm 1 \qquad (\text{ mod } n\ )$$

これを繰り返せば、(ii) でなければ (i) であることがわかる。□

6.3.3 定理

n が素数でなければ、75% 以上の $a \in (\mathbf{Z}/n\mathbf{Z})^\times$ が $(\sharp)$ を満たさない。

6.3.4 証明

$(\sharp)$ を a についての方程式と見做して解の個数を数える。詳細は略。

6.3.5 アルゴリズム

アルゴリズム **6.2.4** の (※) の代わりに $(\sharp)$ を用いる。

合成数 n が $(\sharp)$ のチェックを m 回通過する確率は高々 $\left(\frac{1}{4}\right)^m$ となり、アルゴリズム **6.2.4** より効率が良い。

6.3.6 注

Solovay-Strassen 法や Miller-Rabin 法は誤判定確率が 0 に収束することを使っているので **確率的アルゴリズム** と呼ばれる。確定的アルゴリズムとしては、ガウス和を用いた Adleman-Rumery 法などがある。

7 素因数分解法

7.1 始めに

大きな素因数しかもたない (奇) 数 n を素因数分解したいとき、「小さい素数から順番に割ってみる」のでは絶望的である。

7.1.1 アイデア

$1 < (a, n) < n$ を満たす a にめぐり会うアルゴリズムを作りたい。

$1 < (a, n) < n$ が成り立つと (a, n) は n の真の約数ゆえ、(a, n) がまだ素数でなければ同じアルゴリズムで更に (a, n) の真の約数を求めてゆけば良い。

7.2 Pollard の $p-1$ 法

n が、$p-1$ は小さな素因数しか持たないような素因数 p を持つときは、次の方法によって n の真の約数をみつけることができる。

7.2.1 仮定

p は n の素因数で、$p-1$ の素因数分解を

$$p - 1 = p_1^{e_1} p_2^{e_2} \cdots p_r^{e_r}$$

とする。ある限界値 M (例えば $M = 10^5$) を設定したとき、各 j に対して

$$p_j^{e_j} \leqq M$$

が成り立っていると仮定する。

7.2.2 アルゴリズム

(1) M 以下の全ての素数 q_k に対して $f_k := \lfloor \log_{q_k} M \rfloor$ と置いて

$$K := \prod_{q_k \leqq M} q_k^{f_k}$$

と定める。($q_k^{f_k} \leqq M < q_k^{f_k+1}$ に注意。)

(2) ランダムに a を生成する。

(3) ユークリッドのアルゴリズムによって (a, n) を計算して、$1 < (a, n) < n$ なら終了。

(4) $(a, n) = 1$ ならば $d := (a^K - 1, n)$ を計算する。

(5) $d = n$ ならば (2) へ戻る。$d < n$ ならば $d \geqq p$ なので (後述) 終了。

7.2.3 $d \geqq p$ の証明

作り方より K は各 j について $p_j^{e_j}$ の倍数になっている。従って K は $p-1$ の倍数ゆえ、フェルマーの小定理により

$$a^K \equiv 1 \pmod{p}$$

7.3 2 次ふるい法

7.3.1 アイデア

x を $\sqrt{n}$ の付近で動かすと $x^2 - n$ は比較的小さな数なので、

$$x^2 - n = (\text{ 小さな素数の積 })$$

となる可能性が小さくない。このとき得られる

$$x^2 \equiv (\text{ 小さな素数の積 }) \qquad (\bmod n)$$

という式を組み合わせて

$$X^2 \equiv Y^2 \qquad (\bmod n)$$

という式を導ければ、

$$(X+Y)(X-Y) \equiv 0 \qquad (\bmod n)$$

なので $(X \pm Y, n)$ が n の真の約数となる確率が高い。

7.3.2 パラメータ R, B_1

x は $\lfloor\sqrt{n}\rfloor - R \leqq x \leqq \lfloor\sqrt{n}\rfloor + R$ の範囲を動かす。$x^2 - n$ が B_1 未満の素因数しか持たないときに式を登録する。

7.3.3 例

$n = 101687401$ に対し、パラメータ $R = 1000$, $B_1 = 50$ で実行してみる。

$$\begin{aligned}
9459^2 - n &= -12214720 = -2^6 \times 5 \times 7^2 \times 19 \times 41 \\
9541^2 - n &= -10656720 = -2^4 \times 3^2 \times 5 \times 19^2 \times 41 \\
9581^2 - n &= -9891840 = -2^{12} \times 3 \times 5 \times 7 \times 23 \\
9811^2 - n &= -5431680 = -2^7 \times 3^2 \times 5 \times 23 \times 41 \\
9861^2 - n &= -4448080 = -2^4 \times 5 \times 7 \times 13^2 \times 47 \\
9926^2 - n &= -3161925 = -3^2 \times 5^2 \times 13 \times 23 \times 47 \\
9949^2 - n &= -2704800 = -2^5 \times 3 \times 5^2 \times 7^2 \times 23 \\
9951^2 - n &= -2665000 = -2^3 \times 5^4 \times 13 \times 41 \\
9973^2 - n &= -2226672 = -2^4 \times 3^2 \times 7 \times 47^2 \\
9991^2 - n &= -1867320 = -2^3 \times 3^3 \times 5 \times 7 \times 13 \times 19 \\
10016^2 - n &= -1367145 = -3^3 \times 5 \times 13 \times 19 \times 41 \\
10029^2 - n &= -1106560 = -2^7 \times 5 \times 7 \times 13 \times 19 \\
10043^2 - n &= -825552 = -2^4 \times 3^4 \times 7^2 \times 13 \\
10049^2 - n &= -705000 = -2^3 \times 3 \times 5^4 \times 47 \\
10061^2 - n &= -463680 = -2^6 \times 3^2 \times 5 \times 7 \times 23 \\
10067^2 - n &= -342912 = -2^7 \times 3 \times 19 \times 47 \\
10081^2 - n &= -60840 = -2^3 \times 3^2 \times 5 \times 13^2 \\
10084^2 - n &= -345 = -3 \times 5 \times 23 \\
10096^2 - n &= 241815 = 3 \times 5 \times 7^3 \times 47 \\
10099^2 - n &= 302400 = 2^6 \times 3^3 \times 5^2 \times 7 \\
10124^2 - n &= 807975 = 3^5 \times 5^2 \times 7 \times 19 \\
10133^2 - n &= 990288 = 2^4 \times 3^2 \times 13 \times 23^2 \\
10141^2 - n &= 1152480 = 2^5 \times 3 \times 5 \times 7^4 \\
10199^2 - n &= 2332200 = 2^3 \times 3 \times 5^2 \times 13^2 \times 23
\end{aligned}$$

$$
\begin{aligned}
10225^2 - n = \ \ 2863224 &= 2^3 \times 3^2 \times 7 \times 13 \times 19 \times 23 \\
10349^2 - n = \ \ 5414400 &= 2^9 \times 3^2 \times 5^2 \times 47 \\
10484^2 - n = \ \ 8226855 &= 3^2 \times 5 \times 7^3 \times 13 \times 41 \\
10537^2 - n = \ \ 9340968 &= 2^3 \times 3 \times 7^2 \times 13^2 \times 47 \\
10549^2 - n = \ \ 9594000 &= 2^4 \times 3^2 \times 5^3 \times 13 \times 41 \\
10631^2 - n = 11330760 &= 2^3 \times 3 \times 5 \times 7^2 \times 41 \times 47 \\
10754^2 - n = 13961115 &= 3^2 \times 5 \times 7 \times 23 \times 41 \times 47 \\
10757^2 - n = 14025648 &= 2^4 \times 3 \times 7 \times 13^3 \times 19 \\
10771^2 - n = 14327040 &= 2^8 \times 3 \times 5 \times 7 \times 13 \times 41 \\
11017^2 - n = 19686888 &= 2^3 \times 3^5 \times 13 \times 19 \times 41 \\
11027^2 - n = 19907328 &= 2^8 \times 3 \times 7^2 \times 23^2
\end{aligned}
$$

ここで 1, 2, 20, 21 番目の式の右辺を掛け合わせると全ての数のべきが偶数になることに着目して、

$$a := 9459 \times 9541 \times 10099 \times 10124,$$

$$b := 2^8 \times 3^5 \times 5^3 \times 7^2 \times 19^2 \times 41$$

と置けば

$$(a+b, n) = (9232833075958044, 101687401) = 14533,$$

$$(a-b, n) = (9221554003510044, 101687401) = 6997$$

となる。実際に $6997 = p_{900}$, $14533 = p_{1701}$ はそれぞれ 900 番目、1701 番目の素数で、$n = 6997 \times 14533$ である。

7.3.4 「ふるい」の意味

x を動かす度に小さい素数 p 達で割っていると除算の計算時間が大きくなる。そこで、x で番号付けられた実数の配列を用意して、p 番目ごとの x に $\log p$ を足して行き、最終的に $\log n$ との大小比較で「小さい素数の積かどうか」を判定する。(これが、素数表を作る **エラトステネスの篩** に類似していることから **ふるい法** と呼ばれている。)

7.3.5 関係式の見つけ方

(-1) と小さい素数達のべきを並べて作ったベクトルを有限体 $\mathbf{F}_2$ 上のベクトルと見做して掃き出し法を実行する。自明でない一次結合がゼロベクトルとなったときに、対応する式を掛け合わせれば良い。

7.3.6 注

この他、モダンな素因数分解法には、$x^2 - n$ だけでなく複数の 2 次式を用いる **複数多項式 2 次ふるい法**、代数体の整数環の構造を利用して $X^2 \equiv Y^2 \pmod{n}$ の関係式をみつける **数体ふるい法**、有限体上の楕円曲線の有理点のなす群を利用して $(d, n) > 1$ をみたす d をみつける **楕円曲線法** などがある。

8 有限体の基礎知識

8.1 可換環・体

8.1.1 可換環

加法・減法と可換な乗法が定義されていて、加法の単位元 0 と乗法の単位元 1 を持ち、結合律・分配律などが満たされている集合を **可換環** と言う。

8.1.2 体

0 以外の任意の元が乗法の逆元を持つような可換環を、**体** (たい)、と呼ぶ。

8.1.3 環の乗法群

可換環 R の元のうち、乗法の逆元を持つもの全体の集合を

$$R^{\times} := \{\, x \in R \mid \exists y \in R \text{ s.t. } xy = 1 \,\}$$

(または R^*) と表し、R の乗法群と言う。$R^{\times}$ はアーベル群になる。

8.1.4 体の乗法群

体 F では $F^{\times} = F - \{\, 0 \,\}$ である。

8.1.5 有限体

元の個数が有限個であるような体を **有限体** と言う。

8.2 有限体 F_p

8.2.1 有限体 F_p

素数 p を法とする剰余環 $\mathbf{Z}/p\mathbf{Z} = \{\, 0, 1, 2, \cdots, p-1 \,\}$ は、**1.3.11** によって有限体になる。体であることを強調するためにこれを F_p と書き、p 元体と呼ぶ。

8.2.2 有限体 F_p の演算

F_p の加法・乗法は次式で定められた :

$$F_p \text{ での } a + b := (a + b, \mathbf{mod}\ p), \qquad F_p \text{ での } a \times b := (a \times b, \mathbf{mod}\ p)$$

ただし $(x, \mathbf{mod}\ p)$ は、x を p で割った余りを

$$0 \leqq (x, \mathbf{mod}\ p) \leqq p - 1$$

の範囲で取ることを表す。

また除法は次の様に計算できる。まず、$a \in F_p$ ($a \neq 0$) の逆数は、ユークリッドのアルゴリズムによって $ax + ny = 1$ を満たす整数を求めると x が a の逆元になる。すると、$b \in F_p$ に対して $\dfrac{b}{a} = bx$ となる。

8.2.3 有限体 F_2 の演算表

$+$	0	1
0	0	1
1	1	0

$\times$	0	1
0	0	0
1	0	1

8.2.4 有限体 F_3 の演算表

$+$	0	1	2
0	0	1	2
1	1	2	0
2	2	0	1

$\times$	0	1	2
0	0	0	0
1	0	1	2
2	0	2	1

8.2.5 有限体 F_5 の演算表

$+$	0	1	2	3	4
0	0	1	2	3	4
1	1	2	3	4	0
2	2	3	4	0	1
3	3	4	0	1	2
4	4	0	1	2	3

$\times$	0	1	2	3	4
0	0	0	0	0	0
1	0	1	2	3	4
2	0	2	4	1	3
3	0	3	1	4	2
4	0	4	3	2	1

($2^1 = 2$, $2^2 = 4$, $2^3 = 3$, $2^4 = 1$.)

8.2.6 有限体 F_7 の演算表

$+$	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6
1	1	2	3	4	5	6	0
2	2	3	4	5	6	0	1
3	3	4	5	6	0	1	2
4	4	5	6	0	1	2	3
5	5	6	0	1	2	3	4
6	6	0	1	2	3	4	5

$\times$	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6
2	0	2	4	6	1	3	5
3	0	3	6	2	5	1	4
4	0	4	1	5	2	6	3
5	0	5	3	1	6	4	2
6	0	6	5	4	3	2	1

($3^1 = 3$, $3^2 = 2$, $3^3 = 6$, $3^4 = 4$, $3^5 = 5$, $3^6 = 1$.)

8.3 有限体 F_{p^n}

8.3.1 定理

p を素数、$F_p = \{\, 0, 1, 2, \cdots, p-1 \,\}$ を p 元体、$G(t)$ を F_p 上の n 次の既約多項式とし、集合 F を

$$F := \{\, F_p \text{ 係数の } n-1 \text{ 次以下の多項式 } \}$$

と置く。$G(t)$ を用いて次の様に加法・乗法を定義すると F に有限体の構造を定義することができる。F は p^n 個の元を持つので F_{p^n} と書き表す。

加法 F での $a(t) + b(t) :=$ ($a(t) + b(t)$ を F_p の演算規則で行なったもの)

乗法 F での $a(t) \times b(t) :=$ ($a(t) \times b(t)$ を $G(t)$ で割った余り)

(0 以外の元が乗法の逆元を持つことは次節で述べる。)

8.3.2 有限体 F_4 の演算表

$p=2$, $G(t)=t^2+t+1$ とすると 4 元体 F_4 が得られる。解かり易いように α を F_2 上で $\alpha^2+\alpha+1=0$ なる数とする。

+	0	1	α	α+1
0	0	1	α	α+1
1	1	0	α+1	α
α	α	α+1	0	1
α+1	α+1	α	1	0

×	0	1	α	α+1
0	0	0	0	0
1	0	1	α	α+1
α	0	α	α+1	1
α+1	0	α+1	1	α

($\alpha^1=\alpha$, $\alpha^2=\alpha+1$, $\alpha^3=1$.)

8.3.3 有限体 F_8 の演算表

$p=2$, $G(t)=t^3+t+1$ とすると 8 元体 F_8 が得られる。β を F_2 上で $\beta^3+\beta+1=0$ なる数とする。

+	0	1	β	β+1	β^2	β^2+1	$\beta^2+\beta$	$\beta^2+\beta$+1
0	0	1	β	β+1	β^2	β^2+1	$\beta^2+\beta$	$\beta^2+\beta$+1
1		0	β+1	β	β^2+1	β^2	$\beta^2+\beta$+1	$\beta^2+\beta$
β			0	1	$\beta^2+\beta$	$\beta^2+\beta$+1	β^2	β^2+1
β+1				0	$\beta^2+\beta$+1	$\beta^2+\beta$	β^2+1	β^2
β^2					0	1	β	β+1
β^2+1						0	β+1	β
$\beta^2+\beta$							0	1
$\beta^2+\beta$+1								0

×	0	1	β	β+1	β^2	β^2+1	$\beta^2+\beta$	$\beta^2+\beta$+1
0	0	0	0	0	0	0	0	0
1		1	β	β+1	β^2	β^2+1	$\beta^2+\beta$	$\beta^2+\beta$+1
β			β^2	$\beta^2+\beta$	β+1	1	$\beta^2+\beta$+1	β^2+1
β+1				β^2+1	$\beta^2+\beta$+1	β^2	1	β
β^2					$\beta^2+\beta$	β	β^2+1	1
β^2+1						$\beta^2+\beta$+1	β+1	$\beta^2+\beta$
$\beta^2+\beta$							β	β^2
$\beta^2+\beta$+1								β+1

($\beta^1=\beta$, $\beta^2=\beta^2$, $\beta^3=\beta+1$, $\beta^4=\beta^2+\beta+1$, $\beta^5=\beta^2+\beta$, $\beta^6=\beta^2+1$, $\beta^7=1$.)

8.3.4 有限体 F_{16} の演算表

$p=2$, $G(t)=t^4+t+1$ とすると 16 元体 F_{16} が得られる。

また、$p=2$, $G(t)=t^4+t^3+1$ としても 16 元体 F'_{16} が得られる。

γ を F_2 上で $\gamma^4+\gamma+1=0$ なる数、δ を F_2 上で $\delta^4+\delta^3+1=0$ なる数とすると、対応 $\delta=\gamma^3+1$ によって F_{16} と F'_{16} は同型になる。

8.3.5 有限体 F_9 の演算表

$p=3$, $G(t)=t^2+t+2$ とすると 9 元体 F_9 が得られる。解かり易いように α を F_3 上で$\alpha^2+\alpha+2=0$ なる数とする。

$+$	0	1	2	α	$\alpha+1$	$\alpha+2$	2α	$2\alpha+1$	$2\alpha+2$
0	0	1	2	α	$\alpha+1$	$\alpha+2$	2α	$2\alpha+1$	$2\alpha+2$
1		2	0	$\alpha+1$	$\alpha+2$	α	$2\alpha+1$	$2\alpha+2$	2α
2			1	$\alpha+2$	α	$\alpha+1$	$2\alpha+2$	2α	$2\alpha+1$
α				2α	$2\alpha+1$	$2\alpha+2$	0	1	2
$\alpha+1$					$2\alpha+2$	2α	1	2	0
$\alpha+2$						$2\alpha+1$	2	0	1
2α							α	$\alpha+1$	$\alpha+2$
$2\alpha+1$								$\alpha+2$	α
$2\alpha+2$									$\alpha+1$

$\times$	0	1	2	α	$\alpha+1$	$\alpha+2$	2α	$2\alpha+1$	$2\alpha+2$
0	0	0	0	0	0	0	0	0	0
1		1	2	α	$\alpha+1$	$\alpha+2$	2α	$2\alpha+1$	$2\alpha+2$
2			1	2α	$2\alpha+2$	$2\alpha+1$	α	$\alpha+2$	$\alpha+1$
α				$2\alpha+1$	1	$\alpha+1$	$\alpha+2$	$2\alpha+2$	2
$\alpha+1$					$\alpha+2$	2α	2	α	$2\alpha+1$
$\alpha+2$						2	$2\alpha+2$	1	α
2α							$2\alpha+1$	$\alpha+1$	1
$2\alpha+1$								2	2α
$2\alpha+2$									$\alpha+2$

($\alpha^1=\alpha$, $\alpha^2=2\alpha+1$, $\alpha^3=2\alpha+2$, $\alpha^4=2$, $\alpha^5=2\alpha$, $\alpha^6=\alpha+2$, $\alpha^7=\alpha+1$, $\alpha^8=1$.)

8.4 ユークリッドのアルゴリズム (F_p 上の多項式 version)

8.4.1 ユークリッドのアルゴリズム (F_p 上の多項式 version)

F_p 上の多項式 $a(t)$, $b(t)$ に対して, 次のアルゴリズムによって $a(t)$, $b(t)$ の最大公約因子 $d(t)$ と,

$$a(t)u(t)+b(t)v(t)=d(t)$$

を満たす多項式 $u(t)$, $v(t)$ が求まる。

(1) $b(t)=0$ ならば $d(t):=a(t)$, $u(t):=1$ とせよ。

(2) $b(t)\neq 0$ ならば

a) 次の様に多項式列 $\{r_n(t)\}$, $\{u_n(t)\}$, $\{v_n(t)\}$ を作る：

$$r_0(t):=a(t),\quad r_1(t):=b(t),\quad u_0(t):=1,\quad u_1(t):=0,\quad v_0(t):=0,\quad v_1(t):=1,$$

$$\begin{cases} q(t) := r_{n-2}(t) \text{ を } r_{n-1}(t) \text{ で割った商} \\ r_n(t):=r_{n-2}(t)-q(t)\times r_{n-1}(t)=r_{n-2}(t) \text{ を } r_{n-1}(t) \text{ で割った余り} \\ u_n(t):=u_{n-2}(t)-q(t)\times u_{n-1}(t) \\ v_n(t):=v_{n-2}(t)-q(t)\times v_{n-1}(t) \end{cases} \quad (n=2,3,\cdots).$$

b) $r_n(t) = 0$ となるまで多項式列を計算し、その時点の n で、

$$d(t) := r_{n-1}(t), \qquad u(t) := u_{n-1}(t), \qquad v := v_{n-1}(t)$$

と置け。

(3) $c :=$ ($d(t)$ の最高次の係数), $f :=$ (c の F_p での逆数) として、

$$d(t) := f \times d(t), \qquad u(t) := f \times u(t), \qquad v(t) := f \times v(t)$$

と置け。

8.4.2 注 (バイナリの場合)

バイナリの場合 (F_2 係数の場合) には常に (最高次の係数) $= 1$ であるから最後の (3) は不要になる。

8.4.3 有限体 F_{p^n} の除法

$G(t)$ を F_p 上の n 次既約多項式、

$$F = \{\, F_p \text{ 係数の } n-1 \text{ 次以下の多項式 } \}$$

を $G(t)$ から定まる有限体とする。このとき $a(t) \in F$ ($a(t) \neq 0$) の逆数は、ユークリッドのアルゴリズム (多項式 version) を $a(t), G(t)$ について実行すれば $u(t)$ として求まる。

8.5 有限体に関する定理

8.5.1 定理

有限体は全て第 **8.2** 節、第 **8.3** 節に述べた方法で得られる。

8.5.2 定理

有限体は元の個数が等しければ同型になる。(第 **8.3** 節で $G(t)$ に係わらず F_{p^n} と表したのはその為。)

8.5.3 定理

任意の素数 p と任意の自然数 n に対して p^n元体 F_{p^n} が存在する。言い換えれば F_p-係数の n 次既約多項式が必ず存在する。

8.5.4 定理

有限体 F_{p^n} の乗法群は位数 $p^n - 1$ の巡回群である。すなわち、或る $g \in F^\times$ が存在して、

$$F^\times = \{\, g, g^2, \cdots, g^{p^n-1} = 1 \,\}$$

となる。このような g を F_{p^n} の **原始根** と呼ぶ。

参考文献

1. 岡本龍明・太田和夫:「暗号・ゼロ知識証明・数論」、共立出版 (1995)
2. 情報理論とその応用学会編: 「暗号と認証 = 情報理論とその応用シリーズ 4」、培風館 (1996)
3. 藤崎源二郎・森田康夫・山本芳彦:「数論への出発 = 入門現代の数学 5」、日本評論社 (1980)
4. 和田秀男 :「コンピュータと素因子分解」、遊星社 (1987)
5. D. M. Bressoud: Factorization and Primality Testing = Undergraduate Texts in Mathematics, Springer-Verlag (1989)
6. H. Cohen: A Course in Computational Algebraic Number Theory = Graduate Texts in Mathematics, 138, Springer-Verlag (1993)
7. N. Koblitz: A Course in Number Theory and Cryptography, 2nd Edition = Graduate Texts in Mathematics, 114, Springer-Verlag (1994)
8. A. K. Lenstra, H. W. Lenstra Jr. (Eds.): The developement of the number field sieve = Lecture Notes in Mathematics, 1554, Springer-Verlag (1993)
9. A. Salomaa: Public-Key Cryptgraphy = EATCS Monographs on Theoretical Computer Science, 23, Springer-Verlag (1990) (和訳あり)