# Egisonによる因数分解

梅崎直也@unaoya

株式会社すうがくぶんか

Egison Workshop 2018, 2018 年 11 月 23 日

# $\mathbb{F}_p[x]$ での因数分解

例えば $p = 3$ では

$$
\begin{aligned}
x^3 + x^2 + x + 1 &= (x^2 + 1)(x + 1) \\
x^2 + 2 &= x^2 + 3x + 2 \\
&= (x + 1)(x + 2) \\
x^3 + 1 &= x^3 + 3x^2 + 3x + 1 \\
&= (x + 1)^3
\end{aligned}
$$

# $\mathbb{F}_p[x]$ での演算

多項式の四則、ベキ乗、割り算、gcd、微分などを係数 mod $p$ する。

```
(define $coef-map
  (lambda [$f $P $x]
    (sum' (map2 2#(*' %1 (**' x %2))
    (map f (coefficients P x)) nats0))))

;operation on Fp[x]
(define $coef-mod
  (lambda [$P]
    (coef-map 1#(modulo %1 p) P x)))

(define $p.b.+
  (lambda [%x %y] (coef-mod (+ x y))))
```

# 因数分解アルゴリズムの実装

次の三段階で因数分解を行うアルゴリズム
参考:Wikipedia, Factorization of polynomials over finite fields

1. $f(x)$ を重複度ごとに分解、Square-free factorization
2. 既約因子の次数ごとに分解、Distict-degree factorization
3. 次数成分ごとに既約分解、Cantor-Zassenhaus algorithm

# Square-free factorization

$$f(x) = f_1(x)f_2(x)^2f_3(x)^3\cdots f_n(x)^n$$

と各因子 $f_i(x)$ は重複因子を持たず、それぞれ互いに素であるように分解

$$\begin{aligned} f(x) &= x^{11} + 2x^9 + 2x^8 + x^6 + x^5 + 2x^3 + 2x^2 + 1 \in \mathbb{F}_3[x] \\ &= (x+1)(x^2+1)^3(x+2)^4 \end{aligned}$$

# SFF 実装

$f'$ と $f$ の共通約数が重複因子であることを用いる。ただし標数 $p > 0$ のときは $(x^p)' = 0$ なので注意が必要。

```
;square free factorizatin in ch p
;rewrite p-th power (x^p -> x)
(define $rewrite-rule-p-power
  (lambda [$term]
    (match term math-expr
      {[(* $a ,x^(& ?(divisor? $ p) $k) $r)
        (*' a r (**' x (/ k p)))]
       [_ term]})))

(define $p-inv (map-terms rewrite-rule-p-power $))
```

# SFF 実装

```
(define $step
  (lambda [$i $w $c]
    (let* {[$y (p.gcd w c x)]
           [$fac (fst (p.P./ w y x))]}
      (match y math-expr
        {[,1 {[(fst (p.P./ c y x))] [fac i]}]
         [_ (append (step (+ i 1) y (fst (p.P./ c y x)))
                    {[fac i]})]}))))
(define $SFFp'
  (lambda [$f $x]
    (let* {[$g (p. ∂/∂ f x)]
           [$c (p.gcd f g x)]
           [$w (fst (p.P./ f c x))]
           [$R (step 1 w c)]}
      (match (car R) math-expr {[,1 R]
         [_ (append R (SFFp' (p-inv (car R)) x))]}))))
```

# Distict-degree factorization

$$f(x) = f_1(x)f_2(X)\cdots f_d(x)$$

で $f_i(x)$ の既約因子が $i$ 次式であるように分解する。

$$x^4 + 2 = (x^2 + 2)(x^2 + 1)$$

# DDF 実装

$\mathbb{F}_{q^i}$ の乗法群が位数 $q^i - 1$ の有限群であることから、$x^{q^i} - x$ との公約数を取ればよい。

```
;distinct degree factorization
(define $DDF'
  (lambda [$f $i $x]
    (let* {[$g (p.gcd (- (** x (** p i)) x) f x)]
           [$q (p.monic (fst (P./ f g x)) x)]}
      (match q math-expr
        {[,1 {[f i]}]
         [_ (cons [g i] (DDF' q (+ i 1) x))]}))))

(define $DDF
  (lambda [$f $x] (DDF' f 1 x)))
```

# Cantor-Zassenhaus

重複因子を持たず、既約成分が全て $d$ 次である多項式を因数分解する。

```
(define $Cantor-Zassenhaus
  (lambda [$f $b $x $d]
    (let* {[$m (/ (- (** p d) 1) 2)]
           [$a (p.** b m)]}
      {(p.gcd f a x)
       (p.gcd f (p.+ a 1) x)
       (p.gcd f (p.- a 1) x)})))
```

# 今後の課題

1. 有理数係数での因数分解（Hensel 持ち上げ）
2. 代数的数の扱い（書き換え規則でできそう）
3. 積分
4. いちいち　mod $p$ での演算を実装し直すのが面倒

# 参考文献


1. Wikipedia, Factorization of polynomials over finite fields
2. https://github.com/unaoya/factorization