EUCLIDEAN DIVISION

Let R be a commutative ring. Let $F, G \in R[x]$ s.t. the leading coefficient of G is invertible in R.
Then $\exists!\ (Q,R) \in R[x]^2$ s.t. $F = QG + R$ and $\deg R < \deg G$.

Proof: Uniqueness: If $F = Q_1G + R_1 = Q_2G + R_2$ with $\deg R_1, \deg R_2 < \deg G$ then

$$(Q_1 - Q_2)G = R_2 - R_1 \text{ and } \deg(R_2 - R_1) < \deg G$$
$$\Rightarrow Q_1 = Q_2 \text{ and } R_2 = R_2$$

Existence: We show it by induction on the deg of F.

Base: if $\deg F < \deg G$ $\quad F = 0 \cdot G + F$ ✓

Assume that the statement is true for every $\deg \tilde{F} < \deg F$.

Ind: if $\deg F \geq \deg G$, $\quad F = f_n x^n + \cdots + f_0$

$$G = g_m x^m + \cdots + g_0$$

$f_n \neq 0$, $g_m \neq 0$ and $n \geq m$

Then $\tilde{F} = F - f_n g_m^{-1} x^{n-m} G$ with $\deg \tilde{F} < \deg F$

$\Rightarrow \exists\ \tilde{Q}, \tilde{R} : \tilde{F} = G\tilde{Q} + \tilde{R}$ with $\deg \tilde{R} < \deg G$

Hence $F = \tilde{F} + f_n g_m^{-1} x^{n-m} G$

$$= G\tilde{Q} + \tilde{R} + f_n g_m^{-1} x^{n-m} G$$
$$= \left(\tilde{Q} + f_n g_m^{-1} x^{n-m}\right) G + \tilde{R}. \quad \blacksquare$$

Hence we get the next naive algorithm to divide polynomials

Input: $F, G \in K[X]$ with leading coefficient of G invertible

Output: (Q, R) s.t. $F = QG + R$

$Q = 0$

$R = F$

$m = \deg(G)$

while $\deg(R) \geq m$ do

$n = \deg R$

$s = n - m$

$Q = Q + r_n g_m^{-1} X^s$

$R = R - r_n g_m^{-1} X^s G$

end while

Return (Q, R).

Complexity $\deg F = n - m + 1$ iteration of the "while" and for each iteration performs $O(\deg G)$ operation (multiplying G by a monomial) of type $\times, +, -$.

Hence in the worst case this algorithm does

$O(\deg G(\deg F - \deg G))$

$O(m(n-m))$

The division algorithm can be speed up. In order to do that we have to look at the Newton method for computing the inverse.

Modular inverses using Newton iteration

Let R be a commutative ring with identity. Let $f \in R[X]$ and $\ell \in \mathbb{N}$ s.t. $f(0) = 1$. Compute $g \in R[X]$ s.t.

$$f \cdot g \equiv 1 \mod X^\ell \quad \text{and} \quad \deg g < \ell$$

Proposition: If $\exists\, g \in R[x]$ s.t.

$$(*)\quad fg \equiv 1 \mod x^l \quad \text{and} \quad \deg g < l$$

Then such g is unique.

Proof Let g_1, g_2 be solutions of $(*)$. Then

$$f(g_1 - g_2) \equiv 0 \mod x^l$$

i.e. $f \cdot (g_1 - g_2)$ is a multiple of x^l.

Since $f(0) = 1$, this implies $(g_1 - g_2)(0) = g_1(0) - g_2(0) = 0$

Hence $\exists\, c \in R$ and polynomials h_1, h_2 with deg less than $l-1$ s.t. $g_1(x) = h_1(x) \cdot x + c$ and

$$g_2(x) = h_2(x)\, x + c$$

It follows that $f \cdot (h_1 - h_2)$ is a multiple of x^{l-1}.

By repeating the argument we have $h_1(0) = h_2(0)$.

Then by induction on l we get $g_1 = g_2$.

Remark $(*)$ is an equation in $R[x]/\langle x^l \rangle$, it can be thought as an approximation of a more general problem.

Recall from numerical analysis the Newton iteration and let $\phi(g)$ be an equation that we want to solve, where

$$\phi : \mathbb{R} \longrightarrow \mathbb{R} \text{ is a differentiable function.}$$

From a suitable approximation g_0, the sequence, called Newton iteration step,

$$g_{i+1} = g_i - \frac{\phi(g_i)}{\phi'(g_i)}$$

allows to compute the subsequent approximations and converge towards the desired solution.

In our case $F(g) = \frac{1}{g} - f = 0$

The Newton iteration step is

$$g_{i+1} = g_i - \frac{\frac{1}{g_i} - f}{-\frac{1}{g_i^2}} = 2g_i - f g_i^2$$

The next result tells us a good initial approximation and shows us that this method converges quickly to the solution.

Theorem: Let D be a ring, $f, g_0, g_1, \ldots \in D[x]$ with $f(0) = 1$

$g_0 = 1$, $g_{i+1} \equiv 2g_i - f g_i^2 \mod x^{2^i}$ $\forall i \geq 0$

Then $f g_i \equiv 1 \mod x^{2^i}$ $\forall i$

Proof: Induction on i.

$i = 0$ $f \cdot g_0 = f \cdot 1 \equiv f(0) \cdot 1 \equiv 1 \cdot 1 \equiv 1 \mod x^{2^0}$

$f \cdot g_0 = f(0) \cdot 1 \equiv 1 \mod x$

For the induction step

$$1 - f g_{i+1} \equiv 1 - f(2g_i - f g_i^2) \equiv 1 - 2 f g_i + f^2 g_i^2 \equiv$$

$$\equiv (1 - f g_i)^2 \equiv 0 \mod x^{2^{i+1}}$$ ∎

We get the following algorithm to compute the inverse of $f \mod x^\ell$.

Input $f \in R[x]$ with $f(0) = 1$ and $\ell \in \mathbb{N}$

Output $g \in R[x]$ with $fg \equiv 1 \mod x^\ell$

Algorithm for the inverse of $f \mod x^\ell$

$g_0 = 1$

$r = \lceil \log_2 \ell \rceil$

for $i = 1, \ldots, r$ do

$g_i = 2g_{i-1} - f g_{i-1}^2 \mod x^{2^i}$

end for

return g_r

If $f(0) \neq 1$, we set $f(0)^{-1} =: g_0$ if $f(0)$ is invertible

If $f(0)$ is not invertible then no inverse of f modulo x^ℓ exists since $fg \equiv 1 \bmod x^\ell \Rightarrow f(0)g(0) = 1$ ↯

Example: $f = 3x^2+2x+1 \in \mathbb{F}_7[x]$ $\ell = 4$

$g_0 = 1$

$g_1 \equiv 2g_0 - f g_0^2 = 2 - (3x^2+2x+1) \equiv 5x+1 \bmod x^2$

$g = 2g_1 - f g_1^2 = 2(5x+1) - (3x^2+2x+1)(5x+1)^2$

$= 2x^4+4x^3+x^2+5x+1 \equiv 4x^3+x^2+5x+1 \bmod x^4$

We see that $f \cdot g \equiv 1 \bmod x^4$.

Useful notation

A function $M : \mathbb{N} \longrightarrow \mathbb{N}$ is a polynomial multiplication function for a ring R if

- We can multiply two polynomials of degree $\leq n$ in at most $M(n)$ operations.
- M satisfies $M(n+m) \geq M(n) + M(m)$

For example $n^{\log_2 3}$, $n \log_2 n$

We use this notation in order to state results on complexity independently from the choice of algorithm for multiplying two polynomials.

Thm: The algorithm to find the inverse of f mod x^ℓ is correct.

If $\ell = 2^r$ then it uses at most $3 \cdot M(\ell) + \ell$ arithmetic operations in R ($O(M(\ell))$).

We have seen how to invert $f \bmod x^p$ in $R[X]$ with Newton iteration method.

Recall that we want to find Q and R s.t.

$F = QG + R \quad \deg R < \deg G$.

Assume G monic so that Q, R exist also if R is not a field

Consider $x^m F\left(\frac{1}{x}\right) = \left(x^{n-m} Q\left(\frac{1}{x}\right)\right) \cdot x^m G\left(\frac{1}{x}\right) + x^{n-m+1} x^{m-1} R\left(\frac{1}{x}\right)$

Define $\operatorname{rev}_k(F) = x^k F\left(\frac{1}{x}\right)$ Reversed of F.

We have $F = GQ + R \iff$

$$x^m F\left(\frac{1}{x}\right) = x^m G\left(\frac{1}{x}\right) x^{n-m} Q\left(\frac{1}{x}\right) + x^{n-m+1} \cdot x^{m-1} R\left(\frac{1}{x}\right)$$

$$\Rightarrow x^n F\left(\frac{1}{x}\right) \equiv x^m G\left(\frac{1}{x}\right) \cdot x^{n-m} Q\left(\frac{1}{x}\right) \bmod x^{n-m+1}$$

Using Newton iteration we can give a fast division algorithm in $O(M(n))$ operation where $M(n)$ is the cost of multiplying two polynomials of deg $\leq n$.

$$\operatorname{rev}_n(F) = \operatorname{rev}_m(G) \operatorname{rev}_{n-m}(Q) + x^{n-m+1} \operatorname{rev}_{m-1}(R)$$

$$\Rightarrow \operatorname{rev}_n(F) \equiv \operatorname{rev}_m(G) \operatorname{rev}_{n-m}(Q) \bmod x^{n-m+1}$$

Moreover $\operatorname{rev}_m(G)(0) = 1$ and thus $\operatorname{rev}_m(G)$ is invertible mod x^{n-m+1}

$$\Rightarrow \operatorname{rev}_{n-m}(Q) = \operatorname{rev}_n(F) \cdot \operatorname{rev}_m(G)^{-1} \bmod x^{n-m+1}$$

$$\Rightarrow Q = \operatorname{rev}_{n-m}\left(\operatorname{rev}_{n-m}(Q)\right) \text{ and } R = F - QG$$

Example :

Let $F = 5x^5 + 4x^4 + 3x^3 + 2x^2 + x$ $\quad n = 5$

$G = x^2 + 2x + 3$ $\quad m = 2$

in $\mathbb{F}_7[x]$

$\text{rev}_5(F) = x^4 + 2x^3 + 3x^2 + 4x + 5$

$\text{rev}_2(G) = 3x^2 + 2x + 1$

Claim : $\text{rev}_2(G)^{-1} \equiv 4x^3 + x^2 + 5x + 1 \mod x^4$ in $\mathbb{F}_7[x]$

(To show later)

$\Rightarrow \text{rev}_3(Q) = (x^4 + 2x^3 + 3x^2 + 4x + 5)(4x^3 + x^2 + 5x + 1) \equiv$

$6x^3 + x + 5 \mod x^4$

$\rightsquigarrow Q = 5x^3 + x^2 + 6$

$R = F - QG = 3x + 3$

FAST ALGORITHM FOR DIVISION WITH REMAINDER → Cost $O(M(n))$

Input: R commutative ring with unity

$F, G \in R[x]$, G monic (leading coeff invertible)

Output: $Q, R \in R[x]$ / $F = QG + R$ $\deg R < \deg G$

$n = \deg F$

$m = \deg G$

if $n < m$ then

return $0, F$

else

$G^* = rev_m(G)^{-1} \bmod x^{n-m+1}$ (with Newton inverse method)

$Q^* = G^* \cdot rev_n(F) \bmod x^{n-m+1}$

$Q = x^{n-m} rev_{n-m}(Q^*)$

$R = F - GQ$

return Q, R.

Some recap:

An integral domain R (product of any two nonzero elements is nonzero) with a function $d: R \to \mathbb{N} \cup \{-\infty\}$ is an Euclidean domain if $\forall a, b \in R$, $b \neq 0$ we can divide a and b with remainder: $\exists q, r \in R$ s.t. $a = qb + r$, $d(r) < d(b)$.

We say that $q = a$ quo b (↳ quotient) and $r = a$ rem b (↳ remainder)

d is called Euclidean function.

Ex:
- $R = \mathbb{Z}$, $d(a) = |a|$
- $R = \mathbb{F}[x]$, $d(a) = \deg a$
- $R = \mathbb{Z}[i] = \{a + ib : a, b \in \mathbb{Z}\}$, $d(a+ib) = a^2 + b^2$
- $R = \mathbb{F}$, $d(a) = 1$ if $a \neq 0$, $d(0) = 0$.

Def: Let R be a ring and $a, b, c \in R$. c is greatest common divisor of a, b if $c | a$ and $c | b$, if $d | a$ and $d | b \Rightarrow d | c \; \forall d \in R$.

Extended Euclidean Algorithm:

The Euclidean division allows to define Euclid's algorithm for computing GCD of two polynomials.

Starting from two polynomials (F, G), the extended euclidean algorithm replaces the pair (F, G) with $(G, REM(F, G))$ where $REM(F, G)$ is the remainder of the division, computed with the previous algorithm, until $G = 0$.

The GCD is the last nonzero remainder.

NB This works for every Euclidean ring, with euclidean functio d.

Traditional Euclidean

Algorithm: Input: A, B in an euclidean domain

Output: gcd(A, B)

$R_0 = A$

$R_1 = B$

$i = 1$

while $R_i \neq 0$ do

$R_{i+1} = R_{i-1} \bmod R_i$

$i = i + 1$

end while

Return R_{i-1};

It follows $gcd(F, G) = gcd(R_i, R_{i+1})$ $\forall i$.

If $R_{i+1} = 0$, then $gcd(R_{i+1}, R_i) = R_i$, which shows the correctness.

Bezout Theorem

Thm: If g is the greatest common divisor of two polynomials a and b (not both zero), then there are two polynomials u and v s.t.

$$au + bv = g$$

and either $u=1, v=0$ or $u=0, v=1$ or

$\deg u < \deg b - \deg a$, $\deg v < \deg a - \deg b$.

There is an efficient way of computing the polynomials u and v. This algorithm differs from Euclid's algorithm by few more computations done at each iteration of the loop. Therefore it is called Extended division algorithm

Algorithm (Extended gcd)

Input: A, B univariate polynomials.

Output: u, v, g s.t. $g = au + bv$

$R_0 = A$

$U_0 = 1$

$V_0 = 0$

$R_1 = B$

$U_1 = 0$

$V_1 = +1$

$i = 1$

while $R_i \neq 0$ do

$Q_i = \text{quo}(R_{i-1}, R_i)$

$R = \text{Rem}(R_{i-1}, R_i)$

$U_{i+1} = U_{i-1} - Q_i U_i$; $V_{i+1} = V_{i-1} - Q_i V_i$

$i = i + 1$

end while. return $R_{i-1}, U_{i-1}, V_{i-1}$

Key idea: At each step build U_i, V_i s.t. $R_i = U_i A + V_i B$ ✓

- $i=0$, $U_0 = 1$, $V_0 = 0$
- $i=1$ $\underbrace{0}_{U_1} \cdot A + \underbrace{1}_{V_1} \cdot B = B = R_1$

Performing Euclidean division we get

$$R_{i-1} = Q_i R_i + R_{i+1} \Rightarrow$$

$$R_{i+1} = R_{i-1} - Q_i R_i = \underbrace{(U_{i-1} - Q_i U_i)}_{U_{i+1}} A + \underbrace{(V_{i-1} - Q_i V_i)}_{V_{i+1}} B$$