Polynomial Arithmetic

Stefan Jerg

jerg@in.tum.de

Zentrum Mathematik

Technische Universität München (TUM)

Overview – Polynomial Arithmetic

- Polynomial Arithmetic
  - Generalities
  - Polynomial Addition
  - Polynomial Multiplication
  - Fast Polynomial Algorithms
  - Polynomial Exponentiation
  - Polynomial Substitution
- Polynomial Greatest Common Divisors

Generalities

Assume R is a ring, $a_0,\dots,a_d \in R$

$$P(X) = a_d X^d + a_{d-1} X^{d-1} + \dots + a_0$$

univariate polynomial in X

a_i: coefficients of $P(X)$

$a_i X^i$: monomials / terms of $P(X)$

R: coefficient domain of $P(X)$

$R[X]$: set of all polynomials in X with coefficients in R

$\deg P(X)$: degree of $P(X)$ $\quad (a_d \neq 0 \Leftrightarrow \deg P(X) = n)$

Generalities

Assume R is a ring, $a_0,\ldots,a_d \in R$

$$P(X) = a_d X^d + a_{d-1} X^{d-1} + \ldots + a_0$$

univariate polynomial in X

$lc(P(X))$: leading coefficient - $lc(P(X)) = a_d$

$lt(P(X))$: leading term - $lt(P(X)) = lc(P) X^{\deg P}$

if $lc(P(X)) = 1$, we call $P(X)$ a monic polynomial

Generalities

$$P(X_1,\ldots,X_n) = a_t X_1^{e_{1t}} X_2^{e_{2t}} \cdots X_n^{e_{nt}} + \ldots + a_0 X_1^{e_{10}} X_2^{e_{20}} \cdots X_n^{e_{n0}}$$

multivariate polynomial in $X_1,\ldots,X_n$

$a_i X_1^{e_{1i}} X_2^{e_{2i}} \cdots X_n^{e_{ni}}$: monomials / terms of $P(X)$

$e_{1i} + \ldots + e_{ni}$: total degree of a monomial

$\deg P(X_1,\ldots,X_n)$: maximum of the monomial's total degrees

$R[X_1,\ldots,X_n]$: set of all multivariate polynomials in $X_1,\ldots,X_n$

Generalities

different representations of polynomials

- expanded / recursive representation

$$P_1 = X^2Y^3 + X^2Z + YZ^3 + YZ^2 + Z \qquad \in \mathbb{Z}[X, Y, Z]$$

$$P_2 = \left(Y^3 + Z\right)X^2 + \left(\left(Z^3 + Z^2\right)Y + Z\right) \quad \in \left(\left(\mathbb{Z}[Z]\right)[Y]\right)[X]$$

- variable sparse / variable dense representation

$$P_3 = X^2Y^3Z^0 + X^2Y^0Z + X^0YZ^3 + X^0YZ^2 + X^0Y^0Z$$

$$P_4 = \left(\left(Z^0\right)Y^3 + \left(Z\right)Y^0\right)X^2 + \left(\left(Z^3 + Z^2\right)Y + \left(Z\right)Y^0\right)X^0$$

- degree sparse / degree dense representation

$$P_5 = \left(Z^0Y^3 + 0Y^2 + 0Y^1 + \left(Z^1 + 0Z^0\right)Y^0\right)X^2 + 0X^1 +$$

$$\left(\left(Z^3 + Z^2 + 0Z^1 + 0Z^0\right)Y + \left(Z^1 + 0Z^0\right)Y^0\right)X^0$$

Polynomial Addition

algorithm for polynomial addition using a recursive, variable sparse, degree sparse representation using linear lists

$$e.g.: \quad F(X) = X^7 + 3X^5 - 13X^2 + 3 \triangleq \left(X,(7,1),(5,3),(2,-13),(0,3)\right) = F$$

some short functions we need:

`lt(P), lc(P), le(P)`	:leading term/coefficient/exponent of P
`empty(list)`	:decide whether the list is empty
`iscoef(P)`	:decide whether P is a coefficient
`var(P), terms(P)`	:main variable of P / list of terms of P
`rest(list)`	:the list without the first element
`@, ‘,’`	:concat / create lists

```
PolyCreate (var, terms) := {          //creates a new polynomial
  if empty(terms) then 0
  elif ( le(terms) = 0 ) then lc(terms)
  else (var @ terms)
}
```

Polynomial Addition

```
TermsPlus (FTerms, GTerms) := {      //addition of two term-lists
  if empty(FTerms) then GTerms;
  elif empty(GTerms) then FTerms;
  elif le(FTerms) > le(GTerms)
         then (lt(FTerms) @ TermsPlus(rest(FTerms), GTerms));
  elif le(FTerms) < le(GTerms)
        then (lt(GTerms) @ TermsPlus(rest(GTerms), FTerms));
  else {
        tempc := PolyPlus(lc(FTerms), lc(GTerms));
        if (tempc = 0) then TermsPlus(rest(FTerms), rest(GTerms));
        else
            ((le(FTerms), tempc) @
               TermsPlus(rest(FTerms), rest(GTerms));
  }
}
```

Polynomial Addition

when having different main variables: see one as constant term

$$e.g.: \quad (X+1) + Y \quad = \quad (X+1) + YX^0$$

```
PolyPlus (F, G) := {            //addition of two polynomials
  if iscoef(F) then
        if iscoef(G) then F+G
        else PolyCreate(var(G),TermsPlus(((0,F)),terms(G)));
  elif iscoef(G) then
        PolyCreate(var(F),TermsPlus(((0,G)),terms(F)));
  elif ( var(F) > var(G) ) then
        PolyCreate(var(F),TermsPlus(((0,G)),terms(F)));
  elif ( var(F) < var(G) ) then
        PolyCreate(var(G),TermsPlus(((0,F)),terms(G)));
  else
        PolyCreate(var(F),TermsPlus(terms(F),terms(G)));
}
```

Polynomial Multiplication

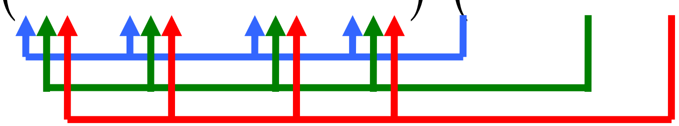
$$F(X) = a_0 X^m + \ldots + a_m$$

$$G(X) = b_0 X^n + \ldots + b_n$$

$$H(X) = F(X)G(X)$$

$$\Rightarrow coef\left(H, X^{m+n-l}\right) = a_l b_0 + a_{l-1} b_1 + \ldots + a_1 b_{l-1} + a_0 b_l$$

e.g.: $\left(X^3 - 3X^2 + 2X - 5\right) \cdot \left(X^2 + X + 2\right) = X^5 - 2X^4 + X^3 - 9X^2 - X$



Polynomial Multiplication

```
PolyTimes (F(X), G(X)) := {
  H := ();
  foreach a_i X^{e_i} in F(X)
     foreach b_k X^{f_k} in G(X)
        coef(H, X^{e_i+f_k}) := coef(H, X^{e_i+f_k}) + a_i*b_k;
  return(H);
}
```

polynomial type	number of terms in H after the i^{th} pass through:	total exponent comparisons
dense	$t+i-1$	$\sum_{i=2}^{t} t+i-1 = t^2 - t + \frac{t(t-1)}{2} \quad \Rightarrow O(t^2)$
sparse	max. $i \cdot t$	$\sum_{i=2}^{t} i \cdot t = \frac{t^2(t+1)}{2} \quad \Rightarrow O(t^3)$

Fast Polynomial Algorithms

use of efficient data structures

e.g. balanced trees or Hash tables

```
TermsPlus2 (FTerms, GTerms) := {
  HTerms := FTerms;
  foreach (e,c) in GTerms {
    oldc := lookup(HTerms, e);
    if ( oldc = () ) then insert((e,c), HTerms);
    else {
        delete(e, HTerms);
        insert((e,PolyPlus(c, oldc)), HTerms);
    }
  }
  return HTerms;
}
```

TermsPlus using data abstraction operators

Fast Polynomial Algorithms

use of efficient data structures

	max. Comparisons	Comparisons	Arith. Ops.
Linear List	k	$O(n)$	$O(n)$
Balanced tree	$\log(k)$	$O(n \log(n))$	$O(n)$
Hash table	$O(1)$	$O(n)$	$O(n)$

Operation counts for addition of polynomials

	max. Comparisons	Comparisons	Arith. Ops.
Linear List	k	$O(n^3)$	$O(n^2)$
Balanced tree	$\log(k)$	$O(n^2 \log(n))$	$O(n^2)$
Hash table	$O(1)$	$O(n^2)$	$O(n^2)$

Operation counts for multiplication of polynomials

Fast Polynomial Algorithms

Divide and Conquer Algorithm

let F, G are polynomials with degree $n = 2^k$

$$F(X) = f_0(X) X^{2^{k-1}} + f_1(X)$$
$$G(X) = g_0(X) X^{2^{k-1}} + g_1(X)$$

$$\deg f_i, \deg g_i \le 2^{k-1}$$

$$F(X)G(X) = f_0 g_0 X^{2^k} + (f_1 g_0 + f_0 g_1) X^{2^{k-1}} + f_1 g_1 =$$
$$= f_0 g_0 X^{2^k} + ((f_1 + f_0)(g_1 + g_0) - f_0 g_0 - f_1 g_1) X^{2^{k-1}} + f_1 g_1$$

Fast Polynomial Algorithms

Divide and Conquer Algorithm

$M(n)$: number of coefficient multiplications

$$\Rightarrow M(n) = 3M\left(\frac{n}{2}\right) = 3^2 M\left(\frac{n}{4}\right) = \ldots = 3^x M\left(\frac{n}{2^x}\right)$$

with $\frac{n}{2^x} = 1$ and $M(1) = 1$

$$\Rightarrow \; M(n) = 3^{\log_2 n} = n^{\log_2 3} \approx n^{1.56496}$$

$\Rightarrow \; O\left(n^{1.56496}\right)$ multiplications

Polynomial Exponentiation

repeated multiplication vers. repeated squaring algorithm

$$(F)^n = (((F \cdot F) \cdot F) \cdot \ldots) \cdot F$$

$$(F)^n = (((F^2)^2)\ldots)^2$$

```
PolyExptRm (P, s) := {
  Q := 1;
  loop for 1 ≤ i ≤ s do {
    Q := P * Q;
  }
  return Q;
}
```

repeated multiplication algorithm

```
PolyExptSq (P, s) := {
  Q := 1; M := P
  loop while s > 0 do {
    if odd(s) then Q := Q * M
    M := M * M;
    s := floor(s/2);
  }
  return Q;
}
```

repeated squaring algorithm

Polynomial Exponentiation

repeated multiplication vers. repeated squaring algorithm

with $P_i = (1 + t_1 + t_2 + \ldots + t_n)^i$ and $L(P_i)$=number of terms in P_i

$$\Rightarrow L(P_i) = \binom{n+i}{n}$$ (proof with induction)

costs of multiplying $\mathrm{P_r}\, P_s$:

$$C_{Sq}(r,s) = L(P_r)L(P_s) = \binom{n+r}{n}\binom{n+s}{n}$$

$$C_{Mul}(r,s) = \sum_{j=r}^{r+s-1} L(P_1)L(P_j) = (n+1)\sum_{j=r}^{r+s-1}\binom{n+j}{n} = (r+s)\binom{n+r+s}{n} - r\binom{n+r}{n}$$

Polynomial Exponentiation

repeated multiplication vers. repeated squaring algorithm

$$\Rightarrow C_{Sq}(r,r) = \binom{n+r}{n}^2 = \ldots = \frac{n^{2r}}{(r!)^2}\left(1 + \frac{r(r+1)}{n}\right) + O\left(n^{2r-2}\right)$$

$$\Rightarrow C_{Mul}(1,2r-1) = 2r\binom{n+2r}{2r} - n - 1 = \ldots = \frac{2r}{(2r)!}n^{2r}\left(1 + \frac{r(2r+1)}{n}\right) + O\left(n^{2r-2}\right)$$

$$\Rightarrow \frac{C_{Sq}(r,r)}{C_{Mul}(1,2r-1)} = \ldots \geq \frac{1}{2r}\binom{2r}{r}\left(1 - \frac{r^2}{n}\right) + O\left(n^{-2}\right) \;\; > 1 \left(r \geq 2, \text{large values of } n\right)$$

Polynomial Exponentiation

algorithm based on the binomial theorem

binomial formula: $(a+b)^n = a^n + na^{n-1}b + \frac{n(n-1)}{2}a^{n-2}b^2 + \ldots + b^n$

let $F(X) = f_d X^d + \left(f_{d-1}X^{d-1} + \ldots + f_0\right)$

set $a := f_d X^d$ and $b := \left(f_{d-1}X^{d-1} + \ldots + f_0\right)$ to receive a fast algorithm

$$\left(f_d X^d + f_{d-1}X^{d-1} + \ldots + f_0\right)^n = f_d^{\,n} X^{d\cdot n} + n\left(f_d X^d\right)^{n-1}\left(f_{d-1}X^{d-1} + \ldots + f_0\right) + \ldots$$

Polynomial Substitution

let $F(X) \in R[X]$, $G \in R$-module *(could also be $\in R[X]$)*

$$\Rightarrow F(X) = f_d X^d + f_{d-1} X^{d-1} + \ldots + f_0$$

$$F(G) = \left(\left(\left(f_d \cdot G + f_{d-1}\right) \cdot G + f_{d-2}\right) \cdot G + \ldots f_1\right) \cdot G + f_0$$

(Horner's rule)

```
TermsHorner (FTerms, G) := {
  H := 0;
  f := le(FTerms);
  foreach (e,c) in FTerms {
    H := G^(f-e) * H + c;
    f := e;
  }
  return H * G^e;
}
```

Overview –Polynomial GCD's

- Polynomial Arithmetic
- Polynomial Greatest Common Divisors
  - Generalities
  - GCD of Several Quantities
  - Polynomial Contents
  - Coefficient Growth
  - Pseudo-Quotients
  - Subresultant Polynomial Remainder Sequence

Generalities

$integral\ domain$: commutative ring with $0 \neq 1$ and no $zero\ divisors$

with $f, g \in R$ integral domain

if $\exists a, h \in R$ with $f = a \cdot h \Rightarrow a, h\ divisors$

if $\exists a, b, h \in R$ with $f = a \cdot h, g = b \cdot h \Rightarrow h\ common\ divisor\ of\ f, g$

$\gcd(f, g)$: greatest common divisor

$\operatorname{lcm}(f, g) := \dfrac{f \cdot g}{\gcd(f, g)}$ least common multiple

$$\gcd(f_1, f_2, \ldots, f_n) = \gcd(f_1, \gcd(f_2, \ldots, f_n)) = \gcd((f_1, \ldots, f_{n-1})\, f_n)$$

Generalities

assume $F, G \in R[X_1, \ldots X_n]$, R *integral domain*

$H_j :=$ *common divisor* (F, G) *with maximal degree in* X_i

$\Rightarrow \operatorname{lcm}(H_i, H_j)$ *is common divisor with maximal degree in* X_i *and* X_j

$\Rightarrow \exists$ *common divisors* (F, G) *with maximal degree in all* X_i

these polynomials we call $\gcd(F, G)$ *up to elements of* R

GCD of several Quantities

$$F_i \in R[X_1,\dots X_n],\ \ R\ \textit{integral domain},\ \ \mathfrak{F} = \{F_1,\dots,F_l\},\ \ H = \gcd(F_1,\dots F_l)$$

$$\gcd(F_1, \gcd(F_2,\dots,F_l)) \ \ \Rightarrow \ \ l-1\ \textit{GCD calculations}$$

more efficient algorithm:

1) $F = F_1 + k_3F_3 + k_5F_5 + \dots,\ \ G = F_2 + k_2F_2 + k_4F_4 + \dots,\ \ (k_i\ \textit{chosen randomly})$
2) $H' := \gcd(F, G)$
3) remove all elements $F_i \in \mathfrak{F}$ with $H' \mid F_i$, add H' to $\mathfrak{F}$
4) repeat choosing new k_i until $|\mathfrak{F}| = 1$

GCD of several Quantities

$\mathrm{res}_x\left(F(X),G(X)\right)$: resultant of F and G

$\mathrm{res}_x\left(F(X),G(X)\right)=0 \Leftrightarrow F,G \; have\ a\ common\ \ factor$

with $G(k)=\mathrm{res}_x\left(F_1,F_2+kF_3\right)$ we get:

$$\gcd\left(F_1,F_2,F_3\right)\neq\gcd\left(F_1,F_2+kF_3\right) \Leftrightarrow k\ is\ zeroes\ of\ \ G(k)$$

$\Rightarrow$ expected $GCD\ calculations$ tends to 1

Polynomial Contents

$$F(X) = f_0 X^n + f_1 X^{n-1} + \ldots + f_n,\ f_0 \neq 0$$

$$\operatorname{cont} F := \gcd(f_0, \ldots, f_n) \quad \text{"}\textit{content of } F\text{"}$$

$$\operatorname{cont} F = 1 \ \Leftrightarrow\ \text{"}\textit{primitive polynomial}\text{"/"}\textit{trivial content}\text{"}$$

$$\operatorname{prim} F := \frac{F(X)}{\operatorname{cont} F} \quad \text{"}\textit{primitive part of } F\text{"}$$

$$\operatorname{cont}_R F := \gcd(\textit{coefficients of monomials of } F) \quad \text{"}\textit{scalar content}\text{"}$$

Polynomial Contents

Proposition (Gauss): If F and G are primitive polynomials over an entire ring R then FG is also primitive.

"Dealing with polynomials over an integral domain the primitive part of a reducible polynomial is still reducible"

Proof: Let $F(X) = f_0 X^m + f_1 X^{m-1} + \ldots + f_m$

$$G(X) = g_0 X^n + g_1 X^{n-1} + \ldots + g_n$$

and $p \in R$ $prime$, $p / \operatorname{cont} FG$

→ …

Polynomial Contents

Proposition (Gauss): If F and G are primitive polynomials over an entire ring R then FG is also primitive.

→ …

the images of F and G in $R_{/(p)}[X]$ are both non zero, since F and G are primitive:

$$\hat{F}(X) = \hat{f}_{m-r} X^{r} + \hat{f}_{m-r-1} X^{r-1} + \ldots + \hat{f}_{m}$$
$$\hat{G}(X) = \hat{g}_{n-s} X^{s} + \hat{g}_{n-s-1} X^{s-1} + \ldots + \hat{g}_{n}$$

with $\hat{f}_{m-r}, \hat{g}_{n-s} \neq 0$

but $\hat{f}_{m-r} \cdot \hat{g}_{n-s} = 0$ since $p / \operatorname{cont} FG$

$$\Rightarrow \textit{contradiction}\ (R \text{ is } \textit{integral domain}!)$$

□

Polynomial Contents

Assume $F, G \in R[X_1, \ldots, X_n]$, $H := \gcd(F, G)$

if X_i only occurs in G, not in F then H not involves X_i

$$\Rightarrow \quad H = \gcd(F, G) = \gcd\left(F, \operatorname{coef}\left(G, X_i^0\right), \operatorname{coef}\left(G, X_i\right), \operatorname{coef}\left(G, X_i^2\right), \ldots\right)$$

$\Rightarrow$ we can remove all variables that do not occur in every polynomial

$$\operatorname{cont} H = \gcd(\operatorname{cont} F, \operatorname{cont} G) = \gcd(f_0, \ldots, f_m, g_0, \ldots, g_m)$$

$\Rightarrow \gcd\left(\frac{F}{\operatorname{cont} H}, \frac{G}{\operatorname{cont} H}\right)$ is primitive (although if F, G are not primitive)

Coefficient Growth

Let $F_1(X), F_2(X)$ be polynomials over a field

$$F_1(X) = \underbrace{q_1(X)}_{quotient} \cdot F_2(X) + \underbrace{F_3(X)}_{remainder} \qquad \deg(F_3) < \deg(F_2)$$

$$F_2(X) = q_2(X) \cdot F_3(X) + F_4(X)$$

$$\vdots$$

$$F_{n-3}(X) = q_{n-3}(X) \cdot F_{n-2}(X) + F_{n-1}(X)$$

$$F_{n-2}(X) = q_{n-2}(X) \cdot F_{n-1}(X) + F_n(X)$$

$$\exists n \in \mathbb{N} : F_{n+1} = 0 \qquad \Rightarrow \qquad F_n = \gcd(F_1, F_2)$$

Coefficient Growth

example 1

$$F_1 = X^8 + X^6 - 3X^4 - 3X^3 + 8X^2 + 2X - 5$$

$$F_2 = 3X^6 + 5X^4 - 4X^2 - 9X + 21$$

$$F_3 = -\frac{5}{9}X^4 + \frac{1}{9}X^2 - \frac{1}{3}$$

$$F_4 = -\frac{117}{25}X^2 - 9X + \frac{441}{25}$$

$$F_5 = \frac{233150}{19773}X - \frac{102500}{6591}$$

$$F_6 = -\frac{1288744821}{543589225}$$

Coefficient Growth

example 2

$$F_1 = X^4 + X^3 - W$$

$$F_2 = X^3 + 2X^2 + 3WX + 1$$

$$F_3 = (-3W + 2)X^2 + (4W - 1)X - 2W + 1$$

$$F_4 = \frac{27W^3 - 2W^2 - 11W + 3}{9W^2 - 12W + 4} X + \frac{9W^3 - W^2 + 4W + 1}{9W^2 - 12W + 4}$$

$$F_5 = \frac{-729W^7 - 738W^6 - 474W^5 + 725W^4 + 81W^3 - 162W^2 + 68W - 8}{729W^6 - 108W^5 - 590W^4 + 206W^3 + 109W^2 - 66W + 9}$$

Pseudo-Quotients

Euclidean GCD algorithm

Let $F_1(X), F_2(X)$ be polynomials over a ring R, $f_i = \mathrm{lc}(F_i(X))$,
$\delta_i = \deg(F_i) - \deg(F_{i+1})$

$$f_2^{\delta_1+1} F_1(X) = Q(X) \cdot F_2(X) + R(X) \qquad \deg(R) < \deg(F_2)$$

$\Rightarrow$ $R(X)$ always has integral coefficients

Pseudo-Quotients

example 1

$$F_1 = X^8 + X^6 - 3X^4 - 3X^3 + 8X^2 + 2X - 5$$
$$F_2 = 3X^6 + 5X^4 - 4X^2 - 9X + 21$$
$$F_3 = -15X^4 + 3X^2 - 9$$
$$F_4 = 15795X^2 + 30375X - 59535$$
$$F_5 = 1254542875143750X - 1654608338437500$$
$$F_6 = 12593338795500743100931141992187500$$

Euclidean PRS

$$F_1 = X^8 + X^6 - 3X^4 - 3X^3 + 8X^2 + 2X - 5$$
$$F_2 = 3X^6 + 5X^4 - 4X^2 - 9X + 21$$
$$F_3 = -5X^4 + X^2 - 3$$
$$F_4 = 13X^2 + 25X - 49$$
$$F_5 = 4663X - 6150$$
$$F_6 = 1$$

Primitive PRS *(in each step division by* $\operatorname{cont}(F_i)$ *)*

Pseudo-Quotients

polynomial remainder sequences

A sequence of polynomials $F_1, F_2, \ldots F_n \in R[X]$ that satisfies

$$\beta_i F_i(X) = \alpha_i F_{i-2}(X) - F_{i-1}(X) \qquad \deg(F_i) < \deg(F_{i-1})$$

where $\alpha_i, \beta_i \in R$, $q(X)$ is a polynomial over R, is called a *"polynomial remainder sequence"* generated from F_1 and F_2.

with $f_i = \mathrm{lc}(F_i(X))$, $\delta_i = \deg(F_i) - \deg(F_{i+1})$ we declare $\alpha_i = f_{i-1}^{\delta_{i-2}+1}$ so that $\beta_i F_i = \mathrm{prem}(F_{i-1}, F_{i-2})$. ($1 \le \beta_i \le \mathrm{cont}(\mathrm{prem}(F_{i-1}, F_{i-2}))$)

Subresultant Poly. Remainder Seq.

according to Collins and Brown

We choose the β_i using following equalities

$$f_i = \mathrm{lc}\left(F_i\left(X\right)\right),\ \delta_i = \deg\left(F_i\right) - \deg\left(F_{i+1}\right),\ \alpha_i = f_{i-1}^{\delta_{i-2}+1}$$

$$h_2 = f_2^{\delta_1} \quad , \quad h_i = f_i^{\delta_{i-1}} h_{i-1}^{1-\delta_{i-1}} \quad \left(i = 3,\ldots,k\right)$$

$$\beta_3 = \left(-1\right)^{\delta_1+1}, \quad \beta_i = \left(-1\right)^{\delta_{i-2}+1} f_{i-2} h_{i-2}^{\delta_{i-2}} \quad \left(i = 4,\ldots,k+1\right)$$

to get the best GCD algorithm that uses a full polynomial remainder seqence

Subresultant Poly. Remainder Seq.

example 1

$$F_1 = X^8 + X^6 - 3X^4 - 3X^3 + 8X^2 + 2X - 5$$
$$F_2 = 3X^6 + 5X^4 - 4X^2 - 9X + 21$$
$$F_3 = 15X^4 - 3X^2 9$$
$$F_4 = 65X^2 + 125X - 245$$
$$F_5 = 9326X - 12300$$
$$F_6 = 260708$$

Polynomial Arithmetic

Stefan Jerg

jerg@in.tum.de

☺ Thanks for listening ☺