Computer Algebra
Lecture 8

James Davenport

University of Bath

12 September 2018

Polynomial Factorisation

- At the beginning, I asked "Do we really know algorithms?
- Can you factor $f := x^5 - 2x^4 + 8x^3 + 3x^2 + 6x - 4$?"
- Well, of course
  $f \cdot (x-1) = x^6 - 3x^5 + 10x^5 - 5x^3 + 3x^2 - 10x + 4 =$
  $(x^3 - 1) * (x^3 - 3x^2 + 10x - 4)$,
- so $f = (x^2 + x + 1) * (x^3 - 3x^2 + 10x - 4)$,
- and it's not hard to see that both factors are irreducible.
- Hardly an algorithm!
- Can't we work modulo primes p?

There are good algorithms for factoring modulo p [CZ81]

Theorem

Modulo p, all irreducible polynomials of degree d divide $x^{p^d} - x$, and no higher-degree irreducibles do.

Let $f_i = \prod g$: g monic irreducible of degree i dividing f.

Then $f_1 = \gcd(x^p - x, f)$, $f_2 = \gcd(x^{p^2} - x, f/f_1)$, $f_3 = \gcd(x^{p^3} - x, f/f_1 f_2)$ etc.

N.B. Do *not* compute $\gcd(x^{p^n} - x, f)$ the obvious way: rather compute $x^{p^n} \pmod f$ by repeated squaring, and only use Euclid afterwards.

But If f_i has degree ki for $k > 1$ we aren't finished.

Assume p odd. $x^{p^i} - x = x\left(x^{(p^i-1)/2} - 1\right)\left(x^{(p^i-1)/2} + 1\right)$.

Then $\gcd\left(f_i, \left(x^{(p^i-1)/2} - 1\right)\right)$ should split f_i in two.

There are good algorithms for factoring modulo p (continued)

If this isn't enough, $\forall j x^{p^i} - x = (x-j)^{p^i} - (x-j) = (x-j)\left((x-j)^{(p^i-1)/2} - 1\right)\left((x-j)^{(p^i-1)/2} + 1\right)$, which gives us a different split via

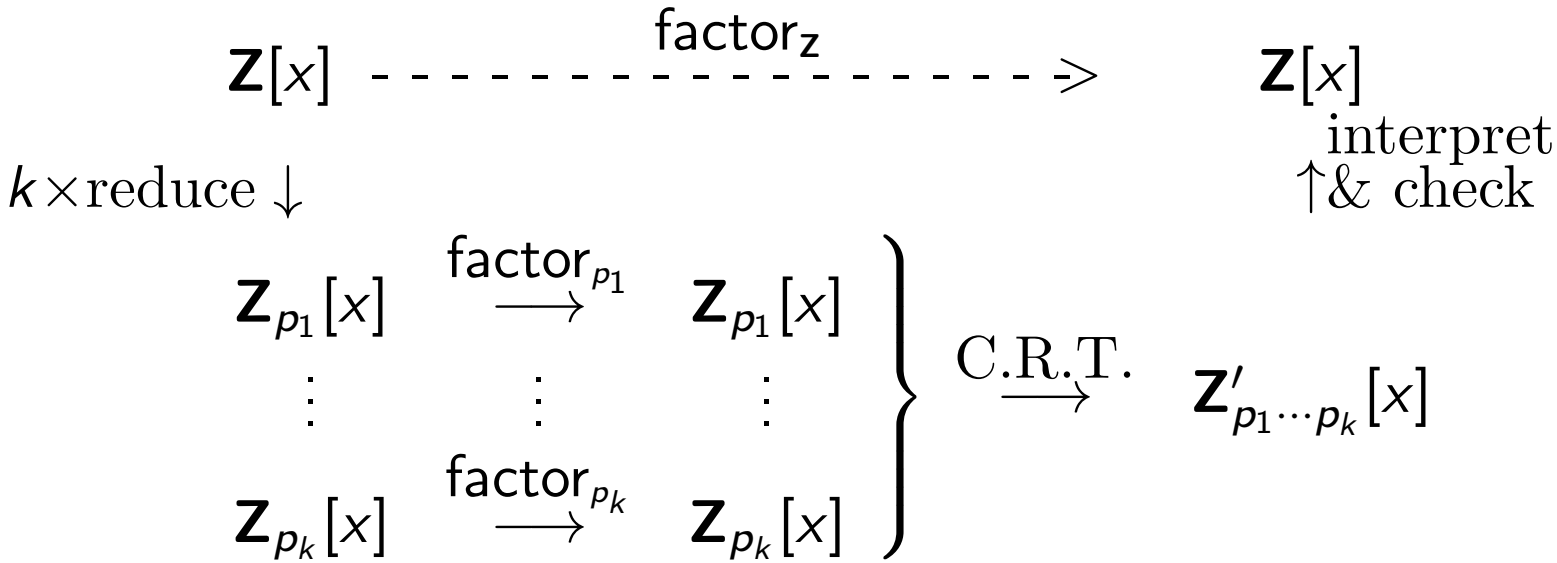
$$\gcd\left(f_i, \left((x-j)^{(p^i-1)/2} - 1\right)\right). \qquad (1)$$

So We can split f_i into factors of degree i, which are irreducible by construction.

This is a *Las Vegas Algorithm*: guaranteed correct, and probably fast: we might be unlucky and have to do several (1).

Onward to C.R.T.?

Figure: Diagram of Many Small Prime factor Algorithm (??)



$\mathbf{Z}'_{p_1\cdots p_k}[x]$ indicates that some of the p_i may have been rejected by the compatibility checks, so the product is over a subset of $p_1 \cdots p_k$.

The questions

1. Are there "good" reductions from R?

factor$_{p_i}$ is not the image of some factor$_{\mathbf{Z}}$ so the "avoid finitely many divide by 0" argument doesn't work.

2. How can we tell if R_i is good?

* Not obvious

3. How many reductions should we take?

* Landau–Mignotte?

4. How do we combine?

Which factor modulo p_1 belongs with which factor modulo p_2 belongs with which factor modulo p_3 . . . ?

5. How do we check the result?

* Not obvious, especially assertions of irreducibility.

These are real issues

- x^4+1 is irreducible, but factors as two quadratics modulo every prime
- Not unique: "Swinnerton-Dyer polynomials" [SD70]
- x^4+3 factors as

$$x^4+3=\left(x^2+2\right)\left(x+4\right)\left(x+3\right) \mod 7; \tag{2}$$

$$x^4+3=\left(x^2+x+6\right)\left(x^2+10\,x+6\right) \mod 11. \tag{3}$$

So really

$$x^4+3=\left(x^2+2\right)\left(x^2+5\right) \mod 7, \tag{4}$$

Do we pair $\left(x^2+x+6\right)$ with $\left(x^2+2\right)$ or $\left(x^2+5\right)$? Both seem feasible, and both are correct. Modulo 77, we do not have unique factorisation.

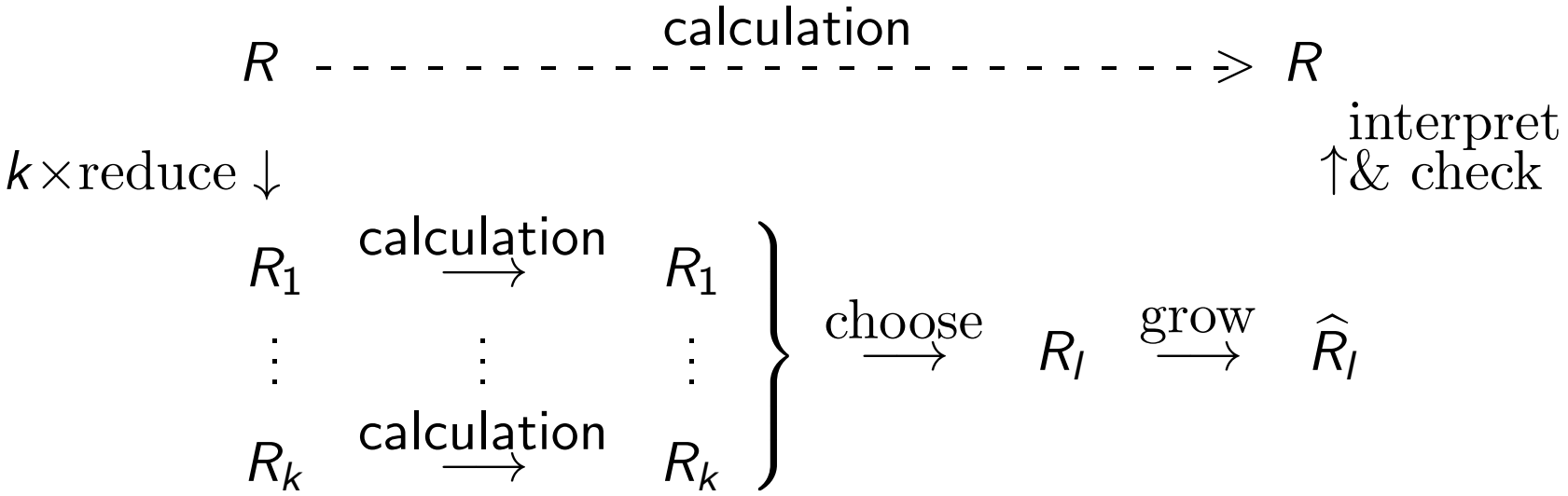
We need a different idea

Write f_p to mean $f \pmod p$ etc., and $f_{p^2} = f_p + pf_2$ etc.
Suppose we are factoring f as $f = gh$ and we know g_p, h_p such that $f_p = g_p h_p \pmod p$. Then
$f_{p^2} = g_{p^2} h_{p^2} = g_p h_p + p(g_p h_2 + h_p g_2) + p^2 g_2 h_2$. Then

$$\frac{f_{p^2} - g_p h_p}{p} \equiv g_p h_2 + h_p g_2 \pmod p \tag{5}$$

(5) is *linear* in the unknowns g_2 and h_2, and can be solved efficiently for unique g_2, h_2 as long as their degrees are less than g_p, h_p. Similarly we can solve from $f_{p^3} = g_{p^3} h_{p^3}$ to get a linear equation in the unknowns g_3 and h_3, and so on.
Eventually we know g_{p^n}, h_{p^n} such that $p^n > 2LM$, where LM is the Landau–Mignotte bound on factors of f, and so g_{p^n} should be g etc.
This process is known as *Hensel Lifting* and lets us go from a solution modulo p to one modulo p^n

Diagrammatic illustration of Hensel Algorithms



where R_l is one of $R_1, \ldots, R_k$.

Square-free Factorisation

Definition

The *derivative* of $f := \sum_{i=0}^{n} a_i x^i$ is $f' := \sum_{i=i}^{n} i a_i x^{i-1}$

Corollary

$(f+g)' = f' + g'$; $(fg)' = f'g + fg'$.

Let $f = \prod_{i=1}^{n} f_i^i$ where the f_i have no common, or repeated, factors.
$\gcd(f, f') = \prod_{i=1}^{n} f_i^{i-1}$, so $g_1 := \frac{f}{\gcd(f,f')} = \prod_{i=1}^{n} f_i$.
$\gcd(g_1, \gcd(f, f')) = \prod_{i=1}^{n} f_i$, so $\frac{g_1}{\gcd(g_1, \gcd(f,f'))}$ is f_1.
$\gcd(f, f') = \prod_{i=1}^{n} f_i^{i-1}$, so the same process will find f_2 etc.
This process is called *Square-free Factorisation* — SQFR

High-level structure of univariate factorisation

procedure FACTORISE($f \in \mathbf{Z}[x]$)
    $\prod_{i=1}^{n} f_i^i$:=SQFR(f)
    Result:=$\emptyset$
    for each nontrivial f_i **do**
        Take a suitable prime p and factorisation of $f_i \pmod p$
        M_i :=Landau–Mignotte(f_i)
        Lift factorisation to be modulo $p^{k_i} > 2M_i$
        Interpret as a factorisation $f_i = \prod_{j=1}^{n_i} f_{i,j}$ over $\mathbf{Z}$,
            combining factors as necessary
        for $j = 1 : n_i$ **do**
            Add (f_j, i) to Result
        end for
    end for
end procedure

Line 5 "Take a suitable prime p" [Mus78]

Example

Suppose f is quartic, and factors modulo p into a linear and a cubic, and modulo q into two quadratics (all such factors being irreducible), we can deduce that it is irreducible over the integers, since no factorization over the integers is compatible with both pieces of information.

[Mus78] suggested one should factor modulo 5 primes and look for compatibility.
Modern mathematics suggests 7 [PPR15, Figure 1]

Univariate Hensel Lifting (Linear Two Factor version)

1: **procedure** $\mathrm{H}(f, g^{(1)}, h^{(1)}, p, k$ with f monic and $\equiv g^{(1)}h^{(1)} \pmod{p})$
2: $g := g^{(1)}$ $h := h^{(1)}$
3: $g^{(r)}, h^{(r)} :=$ EEuclid$(g^{(1)}, h^{(1)})$ in $\mathbf{Z}_p[x]$
4: **for** $i := 2 \ldots k$ **do**
5: $\Delta := \frac{f - gh \pmod{p^i}}{p^{i-1}}$
6: $g^{(c)} := \Delta * h^{(r)} \pmod{(p, g^{(1)})}$
7: $h^{(c)} := \Delta * g^{(r)} \pmod{(p, h^{(1)})}$
8: $g := g + p^{i-1}g^{(c)}$ $h := h + p^{i-1}h^{(c)}$
9: **end for**
10: **return** (g, h)
11: **end procedure**

EEuclid

The extended Euclidean Algorithm.
Instead of just computing $G = \gcd(A, B)$, also compute λ, μ such that $G = \lambda A + \mu B$.
In our case $1 = g^{(1)} g^{(r)} + h^{(1)} h^{(r)}$:
"r" for reciprocal: $g^{(r)} \equiv \frac{1}{g^{(1)}} \pmod{p, h^{(1)}}$.
So need $\gcd(g^{(1)}, h^{(1)})$, which is why we need square-free.

Univariate Hensel Lifting (Linear version) I

1: **procedure** $\mathrm{H}(f, g_1^{(1)}, \ldots, g_n^{(1)}, p, k)$
2: **for** $j := 1 \ldots n$ **do** $g_j^{(1)} := \frac{\mathrm{lc}(f)}{\mathrm{lc}(g_j^{(1)})} g_j^{(1)}$
3: **end for**
4: $F := \mathrm{lc}(f)^{n-1} f$ ▷ leading coefficients imposed
5: **for** $j := 1 \ldots n$ **do**
6: $g_j^{(r)}, h_j^{(r)} := \mathrm{EEuclid}(g_j^{(1)}, \prod_{i \neq j} g_i^{(1)})$ in $\mathbf{Z}_p[x]$
7: **end for**
8: **for** $i := 2 \ldots k$ **do**
9: $\Delta := \dfrac{F - \prod_j g_j \pmod{p^i}}{p^{i-1}}$
10: **if** $\Delta = 0$ **then Break** ▷ True factorization discovered
11: **end if**
12: **for** $j := 1 \ldots n$ **do**
13: $g_j^{(c)} := \Delta * h_j^{(r)} \pmod{(p, g_j^{(1)})}$
14: $g_j := g_j + p^{i-1} g_j^{(c)}$

Univariate Hensel Lifting (Linear version) II

15: **end for**
16: **end for**
17: **for** $j := 1 \ldots n$ **do**
18: $g_j := \mathrm{pp}(g_j)$ ▷ undo the imposed leading coefficients
19: **end for**
20: **return** $(g_1, \ldots, g_n)$
21: **end procedure**

Line 8: Interpret as a factorisation $f_i = \prod_{j=1}^{n_i} f_{i,j}$ over $\mathbf{Z}$

We have a factorisation $f_i = \prod_{j=1}^{n_i} f_{i,j} \pmod{p^{k_i}}$, which might or might not correspond to a factorisation over $\mathbf{Z}$.
Obviously any factors $\pmod{p^{k_i}}$ which are true factors are found and removed. Then what?

1. Consider subsets of $\{f_{i,j}\}$ in turn, seeing if the combination is a factor over $\mathbf{Z}$

Worst case exponential (but pretty hard to trigger); used in practice

2. For each $f_{i,j}$ find $g_{i,j}$ that divides f_i over $\mathbf{Z}$ and is divisible by $f_{i,j} \pmod{p^{k_i}}$ [LLL82]

Polynomial but $O(n^{12})$. This is the origin of "LLL Lattice algorithms"

3. Various more recent ideas [ASZ00, vH02]

Extensions

This all generalises to multivariates, using f_{y-v} instead of f_p, just as modular methods.
Line 4: $F := \mathrm{lc}(f)^{n-1} f$ "leading coefficients imposed"
is a real problem (in the g.c.d. case $n = 2$). Various solutions [Wan78, Zip93].
We can also use this for g.c.d., $G = \gcd(A, B)$ means $A = GH$, so do this $\pmod p$ and lift: this is the EZGCD of [Hea79].
Problems if $\gcd(G, H) \neq 1$, so lift the factorisation of a random combination $\alpha A + \beta B = G(\alpha H + \beta \hat{H}$

Bibliography I


J.A. Abbott, V. Shoup, and P. Zimmermann.
Factorization in $\mathbf{Z}[x]$: The Searching Phase.
In C. Traverso, editor, *Proceedings ISSAC 2000*, pages 1–7, 2000.


E.R. Berlekamp.
Factoring Polynomials over Large Finite Fields.
Math. Comp., 24:713–735, 1970.


D.G. Cantor and H. Zassenhaus.
A New Algorithm for Factoring Polynomials over Finite Fields.
Math. Comp., 36:587–592, 1981.


A.C. Hearn.
Non-Modular Computation of Polynomial Gcd Using Trial Division.
In *Proceedings EUROSAM 79*, pages 227–239, 1979.

Bibliography II


A.K. Lenstra, H.W. Lenstra Jun., and L. Lovász.
Factoring Polynomials with Rational Coefficients.
Math. Ann., 261:515–534, 1982.


D.R. Musser.
On the efficiency of a polynomial irreducibility test.
J. ACM, 25:271–282, 1978.


R. Pemantle, Y. Peres, and I. Rivin.
Four random permutations conjugated by an adversary generate S_n with high probability.
Random Structures & Algorithms, 49:409–428, 2015.


H.P.F. Swinnerton-Dyer.
Letter to E.H. Berlekamp.
Mentioned in [Ber70], 1970.

Bibliography III


M. van Hoeij.
Factoring polynomials and the knapsack problem.
J. Number Theory, 95:167–189, 2002.


P.S. Wang.
An Improved Multivariable Polynomial Factorising Algorithm.
Math. Comp., 32:1215–1231, 1978.


R.E. Zippel.
Effective Polynomial Computation.
Kluwer Academic Publishers, 1993.