# Polynomial Factoring

Ramesh Hariharan

# The Problem

- Factoring Polynomials overs Integers
- Factorization is unique (why?)
- $(x^2 + 5x + 6) \rightarrow (x+2)(x+3)$
- Time: Polynomial in degree

# A Related Problem

- Factoring Integers
- 6 → 2 x 3

Time: No algorithm polynomial in log n is known

If the polynomial is not monic (highest deg coeff=1) then polynomial factorization subsumes integer factorization; so assume that the polynomial is monic

# Another Related Problem

- Factoring Polynomials mod prime p
- Factorization is unique (why?)
- $(x^2 + 1) \rightarrow (x+2)(x+3) \mod 5$

Time: Polynomial in degree

# Yet Another Related Problem

- Factoring Polynomials in number fields
- Factorization is not unique (why?)
- In Q(sqrt(5))
  - 4 = 2.2
  - 4= [3-sqrt(5)] x [3+sqrt(5)]

# Factoring Polynomials

- To factor P(x)
  - Find F(x) such that
    - F(x) has known factorization
    - P(x) divides F(x) but not any of its factors

- GCD of P(x) and one of the irreducible factors of F(x) gives a factor of P(x) in polynomial time

- If P(x) is irreducible then no such F(x) can exist

# Factoring Polynomials mod p

Berlekamp's Algorithm

- The required F(x) can be found in polynomial time!!
- Key Idea:
  - x^p – x = x (x-1) (x-2) … (x-p+1)   mod p
  - f(x)^p – f(x) = f(x) (f(x)-1) (f(x)-2) … (f(x)-p+1)  mod p
  - So F(x)=f(x)^p – f(x) has known factorization mod p by Fermat's theorem

# Factoring Polynomials mod p

Berlekamp's Algorithm

Find f(x) such that

- P(x) divides f(x)^p – f(x)
  - hard
- P(x) does not divide f(x) – i for all i in 0.. p-1
  - easy, keep degree of f(x) smaller than that of P(x)

# Factoring Polynomials mod p

Berlekamp’s Algorithm

Find f(x) such that

- $n = \deg(P(x)) > \deg(f(x))$
- $f(x)^p - f(x) \bmod P(x)$ is 0

# Factoring Polynomials mod p

Berlekamp's Algorithm: Now comes the trick

- f(x) = a + bx + cx^2 ….
- f(x)^p = a + bx^p + c x^2p … , i.e., no cross terms
- f(x)^p–f (x) = a + bx^p + c x^2p … - a + bx + cx^2 …., i.e. degree (n-1)p
- f(x)^p – f(x) mod P(x) = a [1 mod P(x)] + b [x^p mod P(x)] + c [x^2p mod P(x)] … - a + b x + c x^2….
- f(x)^p – f(x) mod P(x) can be represented by a known n-1xn-1 matrix Q-I multiplying the unknown vector v=[a,b,c…] , we solve vQ=0 for v;

# Factoring Polynomials mod p

## Matrix Formulation

$$\underbrace{\begin{pmatrix} a & b & c & \ldots \end{pmatrix}}^{n-1} \left( \overbrace{\begin{pmatrix} x^0 \bmod P(x) \\ x^p \bmod P(x) \\ x^{2p} \bmod P(x) \\ \cdot \\ \cdot \\ x^{(n-1)p} \bmod P(x) \end{pmatrix}}^{n-1} - I \right)$$

# Factoring Polynomials mod p

## Berlekamp's Algorithm: Timing Analysis

- Find Q: n remainder calculations, each poly in n and log p
- Solving v(Q-I) takes poly in n
- Computing gcd of P(x) with each of f(x)-i takes p X poly in n

  can be tweaked to log p * poly in n
- This gives at least one factor, now recurse.
- So time is poly in n and p, improvable to log p

# Factoring Polynomials mod $p^2$

Given the factorization of $P(x)$ mod $p$, can we compute the factorization mod $p^2$

To begin with we have

- $P(x) = A(x) B(x) \bmod p$
- $A,B$ are relatively prime mod $p$
- $A,B$ are monic and therefore $\deg(P)=\deg(A)+\deg(B)$

Let $P(x) = A(x)B(x) + p\, e(x) \bmod p^2$ , where $\deg(e)<\deg(P)$

# Factoring Polynomials mod p^2

Find A′,B′ such that

$$P(x) = (A(x) + pA'(x))\ (B(x) + pB'(x)) \mod p^2$$
$$= A(x)B(x) + p\,(A(x)B'(x)+A'(x)B(x)) \mod p^2$$
$$= P(x) + p\,(A(x)B'(x)+A'(x)B(x) - e(x)) \mod p^2$$

We want

$$A(x)B'(x) + B(x)A'(x) = e(x) \mod p$$

# Factoring Polynomials mod p^2

To find A′,B′ such that A(x)B′(x) + B(x)A′(x) = e(x) mod p

- Since A,B are rel. prime mod p, there exists s<B,t<A such that
  A(x)s(x)+B(x)t(x)=1 (mod p)

- Set B′(x)=e(x)s(x) mod p, A′(x)=e(x)t(x) mod p!!

- Problem: B+pB′ and A+pA′ need not be monic

- Fix: Make deg(B′)<deg(B)
  B′(x)= remainder r(x) of e(x)s(x) wrt B(x) mod p,
  e(x)s(x)=q(x)B(x)+r(x) mod p
  A′(x)=e(x)t(x) + A(x)q(x) mod p

# Factoring Polynomials mod p^2

We also need A+ pA′, B+pB′ to be relatively prime mod p^2 to continue this process

We want s′<B, t′<A such that

- (s+ps′) (A + pA′) + (t+pt′) (B + pB′) = 1 mod p^2

Let As+Bt=1+ p f mod p^2, we want

- 1+ pf + s′ pA +t′ pB + spA′ +tpB′ = 1 mod p^2
- f + s′A+t′B+sA′+tB′ = 0 mod p
- s′A + t′ B = f′ mod p where f′ = - (f+sA′+tB′)

Set s′ = sf′ mod p, t′=tf′ mod p

- Same problem as before so set s′ to remainder of sj wrt B′ mod p and adjust t′ accordingly

# Factoring Polynomials mod p^2

Hensel Lifting

- Given A,B, finding s,t using GCD computation is polynomial in n and log p
- Finding A’,B’ requires finding remainders wrt polynomials of degree n modulo p, so polynomial in n and log p
- Finding s’,t is similar
- In general
  - A factorization of P(x)=A(x)B(x) mod p with A,B relatively prime can be lifted to a factorization P(x)=A’(x)B’(x) mod p^k with A’,B’ rel. prime in time poly in n, k, log p

# Factoring Polynomials on Integers

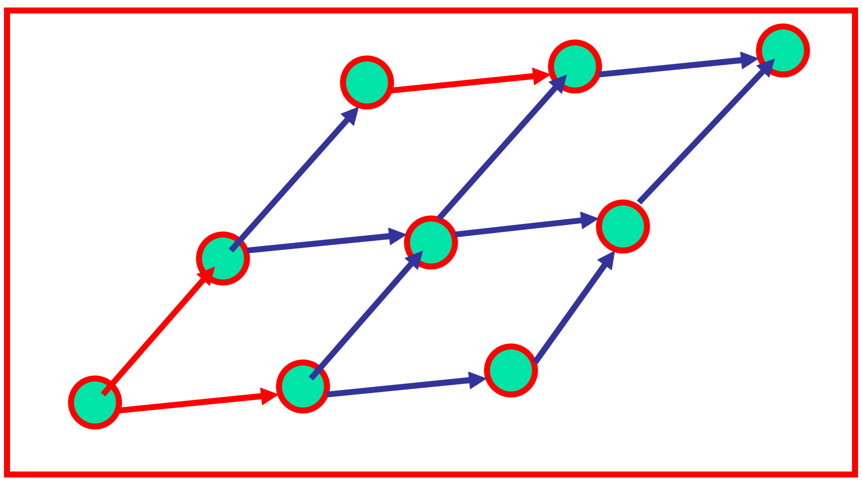
The Algorithm

- Find $P(x)=A(x)B(x) \bmod p$ for some prime $p \sim n$
- Use Hensel lifting to lift so we have $P(x) = A(x)B(x) \bmod p^k$ for some large enough $k$
- Then what? Somehow need to keep coefficients small to avoid wraparound. Needs another new idea.

# Lattices

Given a set of vectors

the lattice generated by these vectors is the set of all integer linear combinations of these vectors



# Polynomials and Lattices

The set of all polynomials of degree at most 2m divisible by polynomial f(x)= sum a_i x^i, deg(f)<m can be represented as a lattice.

- n rows of this matrix generate the lattice in 2m dimensions.
- Multiplying by a row vector of length m (i.e., a polynomial of degree at most m) gives an element of the lattice, i.e., polynomial divisible by f and with degree at most 2m.

2m

m

$$\begin{array}{cccc:cccc} a_0 & a_1 & a_2 & \cdots & a_m & 0 & \cdots & 0 \\ 0 & a_0 & a_1 & \cdots & a_{m-1} & a_m & \cdots & 0 \\ & & \ddots & & . & . & . & . \\ 0 & 0 & . & & . & . & . & a_m \end{array}$$

# The Resultant

Given a(x) of degree m and b(x) of degree n, how does one capture all polynomials which are obtained by taking s(x)a(x) + t(x)b(x), deg(s)<deg(b), deg(t)<deg(a)

- m+n * m+n matrix, premultiply with [s0 s1…sn t0 t1…tm]
- The determinant of this matrix is the resultant(a,b)

$$\begin{vmatrix} a_0 & a_1 & a_2 & \dots & a_m & 0 & \dots & 0 \\ 0 & a_0 & a_1 & \dots & a_{m-1} & a_m & \dots & 0 \\ & & \ddots & . & . & . & . & . \\ 0 & 0 & . & . & . & . & . & .a_m \\ b_0 & b_1 & b_2 & \dots & b_n & 0 & \dots & 0 \\ 0 & b_0 & b_1 & \dots & b_{n-1} & b_n & \dots & 0 \\ & & \ddots & . & . & . & . & . \\ 0 & 0 & . & . & . & . & . & .b_m \end{vmatrix}$$

# A Key Property

Given a(x) of degree m and b(x) of degree n, if a(x) and b(x) are relatively prime then

- there exist s,t such that sa+tb = Resultant(a,b) != 0

  (why not 1? We're working on integers)
- Resultant < |a|^n |b|^m

$$
\begin{vmatrix}
a_0 & a_1 & a_2 & \dots & a_m & 0 & \dots & 0 \\
0 & a_0 & a_1 & \dots & a_{m-1} & a_m & \dots & 0 \\
 & & \ddots & . & . & . & . & . \\
0 & 0 & . & . & . & . & . & .a_m \\
b_0 & b_1 & b_2 & \dots & b_n & 0 & \dots & 0 \\
0 & b_0 & b_1 & \dots & b_{n-1} & b_n & \dots & 0 \\
 & & \ddots & . & . & . & . & . \\
0 & 0 & . & . & . & . & . & .b_m
\end{vmatrix}
$$

# Back to Factorization

Start with P(x) of degree n.
We have found monic, non-constant A(x) of degree <n which divides P(x) mod p^k

- Suppose we find a "short" polynomial B(x) of degree m<n in the lattice generated by A(x) mod p^k (so A divides B mod p^k)
- Short means that Resultant(P,B)<|P|^m |B|^n < p^k
- Then P,B must have a common non-trivial factor
  - if not then there exist s,t such that sP+tB=Resultant(P,B) !=0
  - Then sP+tB=Resultant(P,B) mod p^k
  - A divides Resultant(P,B) mod p^k
  - Resultant(P,B)=0 mod p^k
  - Resultant(P,B)=0
  - Contradiction
- GCD(P,B) gives a factor of P

# How Short Must B be

Short means that Resultant(P,B)$<|P|^m |B|^n < p^k$

- Suppose the entries in P are at most $2^n$, then $|P|^n = 2^{(n^2)}$, we can choose $p^k$ to be larger than this, time is poly in k and log p, so still ok.
- The problem is $|B|^n$; entries in B can be as big as $p^k$.
- We need to keep the entries in B smaller than $p^{k/n}$. Indeed, $|B|$ can be kept down to $|P|\ 2^n$, so $|B|^n$ becomes independent of $p^k$.
- Finding short vectors in lattices in polynomial time requires the LLL algorithm (another talk).

# Thank You